

(19)



JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **10247183 A**(43) Date of publication of application: **14.09.98**

(51) Int. Cl.

G06F 15/16
G06F 9/46
// G06F 13/00

(21) Application number: **09346401**(22) Date of filing: **16.12.97**(30) Priority: **23.12.96 US 98 780015**(71) Applicant: **INTERNATL BUSINESS MACH
CORP <IBM>**

(72) Inventor:
MARSHA LYNN BRANT
KENNETH EDGAR BROWN
PARNELL JAMES DYKES
ERIC DWAIN LINDBERGH
DIANE IREIN OLSON
JEFFERY EDWARD SELDEN
DEVON DANIEL SNYDER
JAMES ORIN WALTZ

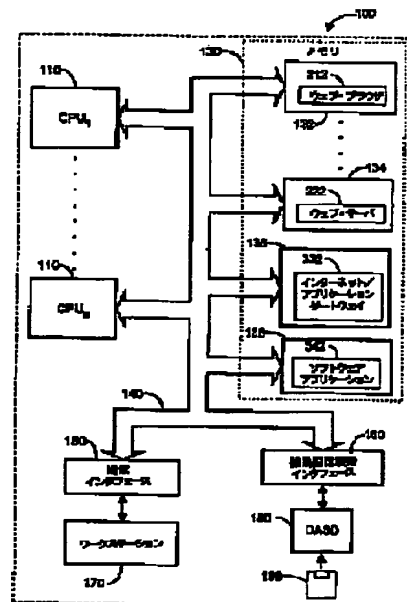
(54) COMPUTER SYSTEM AND METHOD

(57) Abstract:

PROBLEM TO BE SOLVED: To easily access many different application programs in a WWW via a standardized GUI by transmitting and receiving the data to and from plural web browsers and performing the communication between these browsers and a software application.

SOLUTION: The data are inputted via a web browser 212, and these input data are communicated to a web server application 222. The application 222 transfers the proper input data to an application gateway 332 to authenticate and also uniquely identify the browser 212. The gateway 332 decides whether the user of the browser 212 is permitted to access a software application 342. Then the gateway 332 can access the application 342.

COPYRIGHT: (C)1998,JPO



(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平10-247183

(43) 公開日 平成10年(1998) 9月14日

(51) Int.Cl.⁶

識別記号

F I

G 0 6 F 15/16

3 7 0

G 0 6 F 15/16

3 7 0 N

9/46

3 6 0

9/46

3 6 0 B

// G 0 6 F 13/00

3 5 4

13/00

3 5 4 Z

審査請求 未請求 請求項の数27 O L (全 43 頁)

(21) 出願番号

特願平9-346401

(22) 出願日

平成9年(1997)12月16日

(31) 優先権主張番号

0 8 / 7 8 0 0 1 5

(32) 優先日

1996年12月23日

(33) 優先権主張国

米国 (U S)

(71) 出願人 390009531

インターナショナル・ビジネス・マシーンズ・コーポレーション

INTERNATIONAL BUSINESS MACHINES CORPORATION

アメリカ合衆国10504、ニューヨーク州アーモンク (番地なし)

(72) 発明者 マーシャ・リン・プラント

アメリカ合衆国55901 ミネソタ州ロチェスターフォーティサード・ストリートノースウエスト 1902

(74) 代理人 弁理士 坂口 博 (外1名)

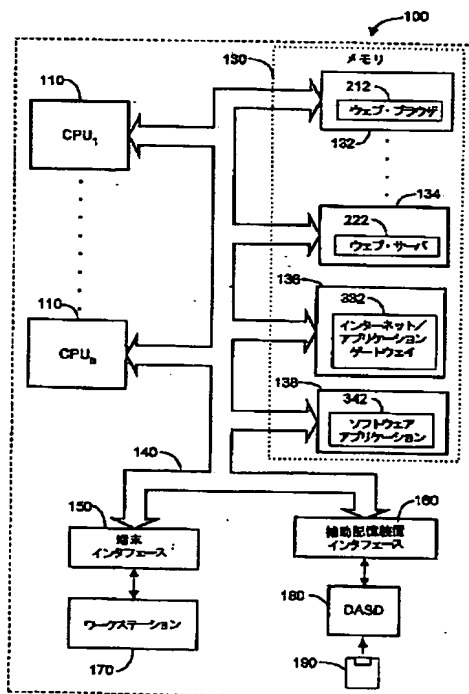
最終頁に続く

(54) 【発明の名称】 コンピュータシステム及び方法

(57) 【要約】

【課題】 本発明の目的は、WWW上で共通ユーザ・インターフェースを介して多数の異なるアプリケーション・プログラムに簡単にアクセスする能力を提供することである。

【解決手段】 WWW上でソフトウェア・アプリケーションにアクセスするための標準的な手順、ルーチン、ツール及びソフトウェア「フック」を提供することによって、ソフトウェア開発者が、アプリケーション・プログラムの機能性に努力を集中し、HTMLを使用してアプリケーション・プログラム用のGUIインターフェースを簡単に提供できるようになる。



【特許請求の範囲】

【請求項1】少なくとも1つの中央処理装置（CPU）と、
CPUに結合されたメモリと、
メモリ内に常駐し、少なくとも1つのCPUによって実行され、共通ユーザ・インターフェースを介して複数のウェブ・ブラウザへまたはこれらからデータを送受する能力を有し、データの識別及び追跡のために識別子機構を使用する、トランザクション・サポート機構とを含む、ワールド・ワイド・ウェブ上で複数のウェブ・ブラウザとソフトウェア・アプリケーションとの間で通信するための共通ユーザ・インターフェースを提供するコンピュータ・システム。

【請求項2】さらに、セキュリティ機構を含み、セキュリティ機構が、メモリ内に常駐し、少なくとも1つのCPUによって実行され、セキュリティ機構が、ソフトウェア・アプリケーションと複数のウェブ・ブラウザとの間に結合され、ソフトウェア・アプリケーションと複数のウェブ・ブラウザとの間のインターフェースを提供し、セキュリティ機構が、複数のウェブ・ブラウザからユーザ入力を受け取り、セキュリティ機構が、受け取った入力に対応するソフトウェア・アプリケーションの認証パラメータを取り出す、請求項1のコンピュータ・システム。

【請求項3】さらに、インターフェース機構を含み、インターフェース機構が、少なくとも1つの変数を処理するためのゲートウェイ機構を含み、ゲートウェイ機構が、メモリ内に常駐し、少なくとも1つのCPUによって実行され、ゲートウェイ機構が、ソフトウェア・アプリケーションの再プログラミングを必要としない、複数のウェブ・ブラウザとソフトウェア・アプリケーションとの間の通信のための汎用共通ゲートウェイ・インターフェースを含む、請求項1のコンピュータ・システム。

【請求項4】さらに、切断機構を含み、切断機構が、メモリ内に常駐し、少なくとも1つのCPUによって実行され、切断機構が、ソフトウェア・アプリケーション・プロセスが再開される時にデータを取り出すことができるように、ソフトウェア・アプリケーション・プロセスが中止される時に、複数のウェブ・ブラウザのうちの1つとソフトウェア・アプリケーション・プロセスとの間の会話のそれぞれに関連する状態データ及び会話識別子を記憶する、請求項1のコンピュータ・システム。

【請求項5】メモリ内に常駐し、少なくとも1つのCPUによって実行され、ソフトウェア・アプリケーションと複数のウェブ・ブラウザとの間に結合され、ソフトウェア・アプリケーションと複数のウェブ・ブラウザとの間のインターフェースを提供し、複数のウェブ・ブラウザからユーザ入力を受け取り、受け取った入力に対応するソフトウェア・アプリケーションの認証パラメータを取り出す、セキュリティ機構と、

少なくとも1つの変数を処理するためのゲートウェイ機構を含み、ゲートウェイ機構が、メモリ内に常駐し、少なくとも1つのCPUによって実行され、ゲートウェイ機構が、ソフトウェア・アプリケーションの再プログラミングを必要としない、複数のウェブ・ブラウザとソフトウェア・アプリケーションとの間の通信のための汎用共通ゲートウェイ・インターフェースを含む、インターフェース機構と、

メモリ内に常駐し、少なくとも1つのCPUによって実行され、ソフトウェア・アプリケーション・プロセスが再開される時にデータを取り出すことができるように、ソフトウェア・アプリケーション・プロセスが中止される時に、複数のウェブ・ブラウザのうちの1つとソフトウェア・アプリケーション・プロセスとの間の会話のそれぞれに関連する状態データ及び会話識別子を記憶する、切断機構とをさらに含む、請求項1のコンピュータ・システム。

【請求項6】トランザクション・サポート機構が、さらに、ソフトウェア・アプリケーションへのネイティブ・インターフェースと通信するための機構を含む、請求項1のコンピュータ・システム。

【請求項7】トランザクション・サポート機構が、ウェブ・サーバ・アプリケーション及びソフトウェア・アプリケーションと通信するアプリケーション・ゲートウェイを含み、アプリケーション・ゲートウェイが、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行され、アプリケーション・ゲートウェイが、識別子機構を含み、識別子機構が、複数のウェブ・ブラウザのそれぞれのために識別子を生成し、ソフトウェア・アプリケーションからのデータを、複数のウェブ・ブラウザのうちの識別子に対応する選択された1つに経路指定する、請求項1のコンピュータ・システム。

【請求項8】アプリケーション・ゲートウェイが、複数のウェブ・サーバから受け取るデータを処理し、アプリケーション・プログラムから受け取るデータを処理する、請求項7のコンピュータ・システム。

【請求項9】ソフトウェア・アプリケーションが、プロセス・エンジニアリング・ソフトウェア・アプリケーションである、請求項1のコンピュータ・システム。

【請求項10】さらに、ソフトウェア・アプリケーションの指示の下で実行される少なくとも1つのアクティビティ・プログラムと通信する少なくとも1つのアクティビティ・プログラム・インターフェース（API）を含み、少なくとも1つのアクティビティ・プログラム・インターフェースが、少なくとも1つのアクティビティ・プログラムとアプリケーション・ゲートウェイとの間で通信する、請求項1のコンピュータ・システム。

【請求項11】複数の中央処理装置（CPU）と、複数のCPUに結合されたメモリと、

それぞれがメモリ内に常駐し、複数のCPUのうちの少

なくとも1つによって実行される、複数のウェブ・ブラウザと、

複数のウェブ・ブラウザのうちの少なくとも1つと通信する、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、ウェブ・サーバ・アプリケーションと、

メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、ソフトウェア・アプリケーションと、

ウェブ・サーバ・アプリケーション及びソフトウェア・アプリケーションへのネイティブ・インターフェースと通信する、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、アプリケーション・ゲートウェイを含み、アプリケーション・ゲートウェイが、複数のウェブ・ブラウザのそれぞれのために識別子を生成し、ソフトウェア・アプリケーションからのデータを、複数のウェブ・ブラウザのうちの識別子に対応する選択された1つに経路指定する、識別子機構を含むワールド・ワイド・ウェブ上でウェブ・ブラウザとソフトウェア・アプリケーションとの間で通信するための共通ユーザ・インターフェースを提供するコンピュータ・システム。

【請求項12】アプリケーション・ゲートウェイが、ウェブ・サーバ・アプリケーション及びアプリケーション・プログラムから受け取るデータを処理する、請求項11のコンピュータ・システム。

【請求項13】ソフトウェア・アプリケーションが、プロセス・エンジニアリング・ソフトウェア・アプリケーションである、請求項11のコンピュータ・システム。

【請求項14】さらに、ソフトウェア・アプリケーションの指示の下で実行される少なくとも1つのアクティビティ・プログラムと通信する少なくとも1つのアクティビティ・プログラム・インターフェース（API）を含み、少なくとも1つのアクティビティ・プログラム・インターフェースが、少なくとも1つのアクティビティ・プログラムとアプリケーション・ゲートウェイとの間で通信する、請求項11のコンピュータ・システム。

【請求項15】ウェブ・サーバ・アプリケーションが、複数のウェブ・ブラウザのうちの1つから渡された認証データから、選択されたウェブ・ブラウザがウェブ・サーバ・アプリケーションへのアクセスを許可されるかどうかを判定する認証機構を含み、

ウェブ・サーバが、複数のウェブ・ブラウザから受け取るデータ及びアプリケーション・ゲートウェイから受け取るデータを処理する請求項11のコンピュータ・システム。

【請求項16】ウェブ・ブラウザが、複数のCPUのうちの少なくとも1つによってクライアント・ワークステーション上で実行される、請求項11のコンピュータ・システム。

【請求項17】ウェブ・サーバ・アプリケーションが、複数のCPUのうちの少なくとも1つによって、ウェブ・サーバ・コンピュータ上で実行される、請求項11のコンピュータ・システム。

【請求項18】アプリケーション・ゲートウェイが、複数のCPUのうちの少なくとも1つによって、ウェブ・サーバ・コンピュータ上で実行される、請求項11のコンピュータ・システム。

【請求項19】アプリケーション・ゲートウェイが、複数のCPUのうちの少なくとも1つによって、第1のコンピュータ上で実行される、請求項11のコンピュータ・システム。

【請求項20】ソフトウェア・アプリケーションが、複数のCPUのうちの少なくとも1つによって、第2のコンピュータ上で実行される、請求項11のコンピュータ・システム。

【請求項21】アプリケーション・ゲートウェイが、複数のCPUのうちの少なくとも1つによって、第2のコンピュータ上で実行される、請求項11のコンピュータ・システム。

【請求項22】複数の中央処理装置（CPU）を提供するステップと、

複数のCPUに結合されたメモリを提供するステップと、

複数のCPUのうちの少なくとも1つによって、メモリ内に常駐する複数のウェブ・ブラウザのうちの少なくとも1つを実行するステップと、

メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、ウェブ・サーバ・アプリケーションを提供するステップと、

メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、ソフトウェア・アプリケーションを提供するステップと、

メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、アプリケーション・ゲートウェイを提供するステップと、

認証データ及び環境データをウェブ・サーバ・アプリケーションに送ることによって、複数のウェブ・ブラウザのうちの選択された1つが、ソフトウェア・アプリケーションへのアクセスを開始するステップと、

認証データが選択されたウェブ・ブラウザにウェブ・サーバ・アプリケーションへのアクセスを許可する場合に、環境データを処理するステップと、

処理された環境データをアプリケーション・ゲートウェイに出力するステップと、

選択されたウェブ・ブラウザと、ソフトウェア・アプリケーションによって実行される所望のプロセスとに対応する識別子を生成するステップと、

メモリ内に常駐し、少なくとも1つのCPUによって実行され、ウェブ・ブラウザからユーザ入力を受け取り、

受け取った入力に対応するソフトウェア・アプリケーションの認証パラメータを取り出す、セキュリティ機構を提供するステップと、
メモリ内に常駐し、少なくとも1つのCPUによって実行され、ウェブ・ブラウザとソフトウェア・アプリケーションとの間で変数及びテンプレートを送受する、インターフェース機構を提供するステップと、
メモリ内に常駐し、少なくとも1つのCPUによって実行され、会話が再開されてソフトウェア・アプリケーションによって所望のプロセスが実行される時に状態データを取り出すことができるように、会話が中止される時にウェブ・ブラウザとソフトウェア・アプリケーションとの間の会話に関連する状態及び会話識別子を記憶する、切断機構を提供するステップと、
所望のプロセスを実行した結果を、識別子を有するアプリケーション・ゲートウェイに返すステップと、
識別子に基づいて、複数のブラウザのうちのどれに結果を送らなければならないかを決定するステップと、
アプリケーション・ゲートウェイからウェブ・サーバ・アプリケーションに結果を送るステップと、
識別子に対応する選択された1つのウェブ・ブラウザに、サーバから結果を送るステップとを含む、ワールド・ワイド・ウェブ上でウェブ・ブラウザとソフトウェア・アプリケーションとの間で通信するための共通ユーザ・インターフェースを提供するための、コンピュータ実施される方法。
【請求項23】ソフトウェア・アプリケーションが、プロセス・エンジニアリング・ソフトウェア・アプリケーションである、請求項22の方法。
【請求項24】ウェブ・ブラウザを走行させるクライアント・ワークステーションと、
ウェブ・サーバ・アプリケーションを走行させるウェブ・サーバ・コンピュータと、
アプリケーション・ゲートウェイを走行させる第1コンピュータと、
ソフトウェア・アプリケーションを走行させる第2コンピュータと、
ウェブ・ブラウザとウェブ・サーバ・アプリケーションとの間でデータを伝送できるようにする、ウェブ・ブラウザとウェブ・サーバ・アプリケーションとの間の通信機構と、
ウェブ・サーバ・アプリケーションとアプリケーション・ゲートウェイとの間でデータを伝送できるようにする、ウェブ・サーバ・アプリケーションとアプリケーション・ゲートウェイとの間の通信機構と、
アプリケーション・ゲートウェイとソフトウェア・アプリケーションとの間でデータを伝送できるようにする、アプリケーション・ゲートウェイとソフトウェア・アプリケーションとの間の通信機構と、
ウェブ・ブラウザとソフトウェア・アプリケーションと

の間で変数及びテンプレートを送受する、インターフェース機構と、
ウェブ・ブラウザとソフトウェア・アプリケーションとの間に結合され、ウェブ・ブラウザとソフトウェア・アプリケーションとの間のインターフェースを提供する、セキュリティ機構と、
会話が再開されてソフトウェア・アプリケーションによって所望のプロセスが実行される時に状態データを取り出すことができるように、会話が中止される時にウェブ・ブラウザとソフトウェア・アプリケーションとの間の会話に関連する状態及び会話識別子を記憶する、切断機構と、
ウェブ・ブラウザがワールド・ワイド・ウェブ上でソフトウェア・アプリケーションと通信できるようにする、複数のアプリケーション・プログラミング・インターフェースとを含む、ワールド・ワイド・ウェブ上でウェブ・ブラウザとソフトウェア・アプリケーションの間で通信するための共通ユーザ・インターフェースを提供するためのシステム。
20 【請求項25】ソフトウェア・アプリケーションが、プロセス・エンジニアリング・ソフトウェア・アプリケーションである、請求項24のシステム。
【請求項26】ウェブ・サーバ・コンピュータが、第1コンピュータを含む、請求項24のシステム。
【請求項27】第1コンピュータが、第2コンピュータを含む、請求項24のシステム。
【発明の詳細な説明】
【0001】
【発明の属する技術分野】本発明は、全般的にはワールド・ワイド・ウェブ上での対話に関し、具体的には、ワールド・ワイド・ウェブを介してソフトウェア・アプリケーションへのアクセスを提供するための方法及び装置に関する。
【0002】
【従来の技術】しばしば、1948年のEDVACコンピュータ・システムの開発が、コンピュータ時代の始まりとして引用される。それ以降、コンピュータ・システムは、極度に洗練された装置に発展し、コンピュータ・システムは、多くの異なる設定で見出される可能性がある。コンピュータ・システムには、通常、ハードウェア（たとえば半導体、回路基板など）とソフトウェア（たとえばコンピュータ・プログラム）の組み合わせが含まれる。半導体加工とコンピュータ・アーキテクチャの進歩によってコンピュータ・ハードウェアの性能が高まるにつれて、ハードウェアの高性能を利用するためにより洗練されたコンピュータ・ソフトウェアが発展し、1～2年前に存在したシステムよりはるかに強力な現在のコンピュータ・システムがもたらされた。
【0003】コンピュータ・システムには、通常、コンピュータの基本機能を制御するオペレーティング・シス

テム・ソフトウェアと、オペレーティング・システムの制御下で走行して所望のタスクを実行する1つまたは複数のソフトウェア・アプリケーションが含まれる。たとえば、通常のIBM Personal ComputerではOS/2オペレーティング・システムが走行し、OS/2オペレーティング・システムの制御下で、ユーザは、ワード・プロセッサなどのアプリケーション・プログラムを実行することができる。コンピュータ・システムの能力が高まるにつれて、高性能コンピュータ・システム用に設計されたソフトウェア・アプリケーションは、極度に強力になってきた。

【0004】技術面の他の変化も、コンピュータの使い方に大きく影響してきた。たとえば、コンピュータの広範囲の普及によって、コンピュータが互いに通信できるようになるコンピュータ・ネットワークの開発が促進された。パーソナル・コンピュータ（PC）の導入に伴って、多数の人々がコンピューティングを利用できるようになった。パーソナル・コンピュータ用のネットワークが開発されて、個々のユーザが互いに通信できるようになった。この形で、1企業内の多数の人間が、単一のコンピュータ・システム上で走行するソフトウェア・アプリケーションを用いて、ネットワークを介して同時に通信できるようになった。

【0005】最近非常に人気のある重要なコンピュータ・ネットワークがインターネットである。インターネットは、現代のコンピュータ及びネットワークの普及から発生し、全体として「ワールド・ワイド・ウェブ」またはWWWを構成するウェブ・ページによって互いにリンクされたコンピュータ・システムの洗練された国際ネットワークに発展した。WWWへのアクセスを望む個々のPC（すなわちワークステーション）のユーザは、通常は、ウェブ・ブラウザと称するソフトウェア・アプリケーションを介してこれを行う。ウェブ・ブラウザは、ウェブ・サーバと称する他のコンピュータへのWWWを介する接続を行い、ユーザのワークステーションに表示される情報をウェブ・サーバから受信する。ユーザに表示される情報は、通常は、ハイパーテキスト・マークアップ言語（HTML）と称する特殊な言語を使用して構成される。HTMLを使用するウェブ・ブラウザは、現在、市販されているコンピュータ・システムのほとんどすべてで使用可能であり、コンピュータとモデムにアクセスできる人であれば事実上誰でもWWWにアクセスできるようになっている。WWWは、ますます一般的になりつつあるが、WWWにアクセスするコンピュータ・ユーザの急激な増加が、それに付随する問題をもたらした。これらの問題のいくつかを、以下で示す。

【0006】インターネットとWWWの人気が高まるのに伴って、インターネットが売上と効率の両方を高めるための新しい方法をもたらすことが認識された。ウェブ・ブラウザを有するユーザが、会社のソフトウェア・ア

プリケーションと直接対話できるならば、所与のトランザクションが単純化される。たとえば、誰かが普通にレンタ・カーを予約する方法を検討する。その人は、レンタ・カー代理店に電話し、電話を介して自分の情報（すなわち、氏名、住所、クレジット・カード番号など）をレンタ・カー代理人に知らせる。代理人は、この情報を自動車レンタル・ソフトウェア・アプリケーションに入力して、自動車を予約するプロセスを初期設定しなければならない。ウェブ・ユーザ用のより効率的な自動車予約システムでは、ユーザが自動車レンタル・ソフトウェア・アプリケーションと直接対話できるようになるはずである。これによって、現在レンタ・カー代理人が実行している作業の大半が除去されるはずである。しかし、ウェブ・ユーザと直接対話できる自動車レンタル・ソフトウェア・アプリケーションを案出するには、カスタム・インターフェース・ソフトウェアを作成する必要がある。同様に、WWWを介してアクセスされる異なるソフトウェア・アプリケーションごとに、カスタム・ユーザ・インターフェースを作成しなければならない。好ましいインターフェースは、グラフィカル・ユーザ・インターフェース（GUI）になるはずである。あるソフトウェア・アプリケーション用のカスタムGUIを生成する処理は、時間がかかり、高価であり、通常は、他のソフトウェア・アプリケーションとの通信には使用できない独自ユーザ・インターフェースがもたらされる。これは、会社がWWWを介するソフトウェア・アプリケーションへのアクセスを提供しなくなる大きな阻害要因となることを意味する。

【0007】さらに、多数のコンピュータ・ユーザは、ハードウェアまたはソフトウェア的に非常に異なる形態のコンピュータ・プラットフォームを採用している。たとえば、IBM互換パーソナル・コンピュータは、現在市販されているもっとも一般的なタイプのコンピュータであるが、他社は、現在導入され使用されている非常に多数のコンピュータ・システムを有する非常に異なる製品系列を開発してきた。これらの全く異なるハードウェア・システムでは、通常は完全に異なるオペレーティング・システムが使用される。これらのさまざまなハードウェア・システム及びソフトウェア・システムの存在は、通常は、異なるハードウェア・プラットフォームのそれぞれに所与のソフトウェア・アプリケーション用のカスタムGUIを「移植」または変換するために、完全に新規のプログラミングと開発の労力を必要とする。多くの会社は、最も一般的なハードウェアとソフトウェアの組み合わせだけをサポートし、これによって、市場のシェアが制限され、そのソフトウェア・アプリケーションにアクセスできるユーザの数が少なくなる。

【0008】複数の関連しないユーザ・インターフェースの問題によって、やはりWWWを介するソフトウェア・アプリケーションのすばい採用を阻止する可能性が

あるもう1つの問題が明らかになる。ある会社が、所与のソフトウェア・アプリケーション用のカスタムGUIを開発するコストを吸収したと仮定しても、1つの会社によって開発されたそのGUIは、別の会社によってその会社のソフトウェア・アプリケーション用に開発されたGUIとは非常に異なる可能性が高い。通常、各ソフトウェア・ベンダは、特定のソフトウェア・アプリケーションごとにカスタムGUIを作成し、その結果、ユーザは、WWWを介してソフトウェア・アプリケーションにアクセスする時には必ず、そのソフトウェア・アプリケーションに固有の特徴にアクセスできるようになる。しかし、各製品は異なる特徴を有し、各ベンダはアプリケーション/ユーザ対話に関して異なる標準を有するので、その結果、ユーザが出会うソフトウェア・アプリケーションのそれぞれに非常に異なるユーザ・インターフェースが存在することがしばしばである。ユーザは、アクセスしたいソフトウェア・アプリケーションのそれぞれと対話するための基本的なスキルを「再学習」しなければならないことがしばしばである。

【0009】WWW上でのソフトウェア・アクセスに関する現在の状況は、パーソナル・コンピュータの初期の時代に多少似ている。1980年代初期に、IBMパーソナル・コンピュータ(PC)が導入され、業界標準のハードウェア・プラットフォームとしてすばやく採用された。しかし、ハードウェア・プラットフォームは相対的に標準的であったが、独立ソフトウェア・ベンダのそれぞれが、それぞれのアプリケーション・プログラムのために大きく異なるユーザ・インターフェースを作成した。これが、パーソナル・コンピュータのユーザによる新しいアプリケーション・プログラムのすばやい採用の妨げになった。ユーザは、長い訓練なしに新しいソフトウェア・アプリケーションを効率的に使用することができなかったので、多数の新しいソフトウェア・アプリケーションが採用されなかった。

【0010】

【発明が解決しようとする課題】しかし、現在、新しい標準がパーソナル・コンピュータ・ソフトウェア・アプリケーション用に開発され、採用されており、これによって、新しいソフトウェア・アプリケーションのために行わなければならない再学習の量が劇的に減少した。IBM社のOS/2などの最新技術のGUIオペレーティング・システムでは、共通のユーザ動作の多くが標準化され、独立ソフトウェア開発者及びベンダに、最も標準的なユーザ・インターフェースの構成要素及び特徴にアクセスするのに必要な「フック」またはプログラミング・ツールが提供され、これによって、エンド・ユーザの学習曲線が劇的に低くなった。残念ながら、この標準化の努力は、まだWWWには浸透していない。実際、ソフトウェア・アプリケーションの配布システムとしてWWWを簡単に利用できることが、現在WWW上で入手でき

る全く異なるソフトウェアの急激な増加をつのらせている。さらに、入手可能な最も強力なソフトウェア・アプリケーションのうちの一部は、直観的でもグラフィカルでもないユーザ・インターフェースを有する。新しいアプリケーションごとに新しいインターフェースを再学習する過程は、冗長で時間がかかり、非生産的になる傾向を有する。したがって、現在多数のソフトウェア・アプリケーションがWWW上でアクセス可能ではあるが、これらのさまざまなプログラムにアクセスする方法の学習は、エンド・ユーザにとって、時間がかかり、いらいらさせられ、威嚇的になる可能性がある。

【0011】

【課題を解決するための手段】WWW上でソフトウェア・アプリケーションへのアクセスを提供することの重要性と、既存の解決策の現在の制約の両方を認識して、本発明は、WWW上で標準化されたGUIを介して多数の異なるアプリケーション・プログラムに簡単にアクセスする能力を提供する。WWW上でソフトウェア・アプリケーションにアクセスするための標準的な手順、ルーチン、ツール及びソフトウェア「フック」を提供することによって、ソフトウェア開発者が、アプリケーション・プログラムの機能性に努力を集中し、HTMLを使用してアプリケーション・プログラム用のGUIインターフェースを簡単に提供できるようになる。

【0012】上で述べたように、HTMLは、現在市販されているコンピュータ・システムのほとんどすべてで使用できる周知の言語である。さらに、HTMLは、かなり明確に制御され、標準化された言語なので、新しいソフトウェア・アプリケーション機能は、それが開発され、HTMLによってサポートされた時に追加できる。さらに、HTMLは、広範囲に採用された非独自技術なので、本発明によって、非常に小規模のソフトウェア開発者にも、巨大な市場へのオープン・アクセスを提供できる。さらに、本発明を用いると、ソフトウェア開発者は、標準アクセス・プロトコルを採用でき、これによって、HTMLを認識するブラウザを利用できるコンピュータ・システムのすべてに対するサポートを提供できるようになる。最後に、ユーザ・インターフェース、認証/セキュリティ及びウェブ・トランザクション・サポートの問題に対する、実施しやすく標準化された解決を提供することによって、本発明の共通ユーザ・インターフェースは、以前の解決に存在する限界を克服する。

【0013】本発明の前述及び他の特徴及び長所は、添付図面に図示された、本発明の好ましい実施例の以下の詳細な説明から明白になる。

【0014】

【発明の実施の形態】

概要

ウェブ・トランザクション

ここで図2を参照すると、クライアント・ワークステー

ション210で走行する標準的なウェブ・ブラウザ212と、ウェブ・サーバ・コンピュータ・システム220で走行するウェブ・サーバ・アプリケーション222の間の通常のトランザクションは、接続216を介して発生する。クライアント・ワークステーション210は、ローカル・エリア・ネットワーク(LAN)を介するか他のタイプのコンピュータ・ネットワークまたは他の相互接続を介して、他のコンピュータ・システムに結合することができる。同様に、ウェブ・サーバ・コンピュータ・システム220も、他のコンピュータに結合することができる。クライアント・ワークステーション210は、ウェブ・ブラウザ212を使用することによってWWWへのアクセスを提供できるコンピュータであれば何でもよい。これには、ハンドヘルド・コンピュータ、ポータブル・コンピュータまたはラップトップ・コンピュータ、標準的なデスクトップ・コンピュータ・システム、パーソナル・デジタル・アシスタント(PDA)、メインフレームに接続された非プログラム式端末などが含まれる。

【0015】ウェブ・ブラウザ212は、クライアント・ワークステーション210のユーザが接続216を介して他のコンピュータと通信できるようにする、クライアント・ワークステーション210で走行するソフトウェア・プログラムである。ウェブ・ブラウザ212には、WWW上でデータを送受信する能力を有するウェブ・ブラウザのすべてが含まれる。これには、IBM社のWeb Explorer、Netscape社のNavigator、Microsoft社のInternet Explorer、Apple Computer社のCyberDogなどの市販ソフトウェア・アプリケーションと、WWW上の情報のアクセスまたは処理のための、現在存在するか将来に開発される他のソフトウェア・アプリケーションが含まれる。接続216の好ましい実施態様は、インターネットへの適当な接続であり、これには、ハードワイヤ接続、モデムまたは高速T1回線を介する電話アクセス、赤外線その他の無線通信、コンピュータ・ネットワーク通信(有線または無線)、現在既知であるか将来に開発される他の適当なコンピュータ間接続が含まれる。

【0016】クライアント・ワークステーション210とウェブ・サーバ・コンピュータ・システム220は、物理的または論理的に同一のコンピュータ・システムとすることができることに留意されたい。ウェブ・ブラウザ212は、通常は、クライアント・ワークステーション210のユーザにHTMLデータのページを表示する。他のタイプ(HTML以外)のデータも、ウェブ・ブラウザ212に送信される可能性があり、これには、テキスト・データ、グラフィカル・データ(たとえばGIF(Graphic Image Format)ファイル)、オーディオ・データまたはサウンド・ファイル(たとえばWAVファイル)、ジャバ・アプレット(実行可能コード)、MIME(Multipurpose Internet Mail Extensions)デ

ータ(前述及び他のデータ型の組み合わせが含まれる可能性がある)と称する特殊なデータが含まれる。

【0017】ウェブ・サーバ・アプリケーション222は、クライアント・ワークステーション210のユーザがウェブ・サーバ・コンピュータ・システム220によって制御される情報にアクセスできるようにする、ウェブ・サーバ・コンピュータ・システム220で走行するソフトウェア・プログラムである。本発明によるウェブ・サーバ・アプリケーション222の好ましい実施態様の1つは、IBM社のInternet Connection Serverなどの市販ウェブ・サーバ・アプリケーションである。他のアプリケーションも、本発明との互換性を有する。ウェブ・サーバ・コンピュータ・システム220は、通常は、クライアント・ワークステーション210でユーザが行った動作を反映する、ウェブ・ブラウザ212による要求に応答して、ウェブ・ブラウザ212にHTMLデータのページを出力する。さらに、上で説明したように、ウェブ・サーバ・コンピュータ・システム220は、ウェブ・ブラウザ212に他のタイプのデータを出力することもできる。出力データには、静的HTMLページ(ページの内容が変化しないことを意味する)または、動的に決定され、出力データに挿入されなければならないデータを含めることができる。ウェブ・サーバ・アプリケーション222は、ウェブ・サーバ・コンピュータ・システム220内のメモリからまたは他のコンピュータ・システムから受け取る部分から出力データ(たとえばHTMLページ)を動的に作成することができ、また、以前にまたは別のコンピュータによって開発されたHTMLページまたは他の情報を単純に渡すことができる。

【0018】ウェブ・ブラウザ212は、通常は、接続216を介してウェブ・サーバ・コンピュータ・システム220に入力(たとえばURL(Uniform Resource Locator)やHTMLページ)を送信することによって、ウェブ・サーバ・アプリケーション222と対話する。この入力は、通常はハイパーテキスト転送プロトコル(HTTP)1.0を使用して送信される。ウェブ・サーバ・アプリケーション222が走行するウェブ・サーバ・コンピュータ・システム220は、ウェブ・ブラウザ212から入力を受信し、これに応答して、ウェブ・ブラウザ212にデータ(たとえばHTMLページ)を出力する。上で述べた処理は、インターネットを介する基本的なトランザクションを示すものであり、本発明の範囲に含まれる多数の詳細及び変形は、本発明の概念を理解するための簡単な文脈を提供するために本明細書では開示されないことを理解されたい。

【0019】ウェブ・サーバ・コンピュータ・システム220は、コモン・ゲートウェイ・インターフェース(CGI)モジュールを含む、他の多数のソフトウェア構成要素も有する可能性がある。CGIモジュールは、

ウェブ・サーバ・アプリケーション222と他のソフトウェア・アプリケーションの間のインターフェースとして使用することができる。たとえば、CGIモジュールは、ウェブ・サーバ・アプリケーション222とカレンダー・ソフトウェア・アプリケーションの間のリンクを提供でき、これによって、ウェブ・サーバ・アプリケーション222が、たとえば、ウェブ・ブラウザ212に出力するウェブ・ページに動的なカレンダー情報を挿入できるようになる。したがって、CGIを用いると、ウェブ・サーバが、他のソフトウェア・アプリケーションからの動的データを配布できるようになる。残念ながら、CGIのプログラム作成は、CGIが満足しなければならない仕様が多数あるので、時間のかかる作業である。さらに、一般に、ウェブ・サーバ・アプリケーション222とインターフェースする異なるソフトウェア・アプリケーションごとに、別々のCGIが必要になる。さらに、異なるユーザに同一のソフトウェア・アプリケーションまたはデータへの異なるレベルのアクセス権を与えるなど、異なる機能を実行するためには、異なるCGIが必要になる可能性がある。一般に、CGIの数が増えるにつれて、ウェブ・サーバ・コンピュータ・システム220の性能が低下する。

【0020】ウェブ・ページ

ここで図3を参照すると、ウェブ・ページは、主に、クライアント・ワークステーション210のモニタでの表示を目的とするビジュアル・データである。ウェブ・ページは、一般に、ハイパーテキスト・マークアップ言語（HTML）で記述される。ウェブ・サーバ・コンピュータ・システム220で走行するウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212からのウェブ・ページ要求を受信した時に、HTMLでウェブ・ページを作成するか、事前に作成されたウェブ・ページを含むファイルを取り出し、接続216を介して要求元のウェブ・ブラウザ212に送信する。ウェブ・ブラウザ212は、HTMLを理解し、解釈し、そのウェブ・ページをクライアント・ワークステーション210のモニタに出力する。この結果の、ユーザの画面に表示されるウェブ・ページには、テキスト、グラフィックス及びリンク（他のウェブ・ページのURLアドレス）が含まれる可能性がある。これらの他のウェブ・ページ（すなわち、リンクによって表されるウェブ・ページ）は、同一のウェブ・サーバ上にあっても、異なるウェブ・サーバ上にあってもよい。ユーザは、マウスまたは他のポインティング装置を使用してこれらのリンクをクリックすることによって、これらの他のウェブ・ページを取り出すことができる。全世界の他のサーバ上の他のウェブ・ページへのリンクを有するウェブ・ページのシステム全体が、集散的に「ワールド・ワイド・ウェブ」（WWW）を構成する。

【0021】一部のウェブ・ページは、ウェブ・ブラウ

ザ212からの入力を誘うように設計されている。たとえば、ウェブ・ページのHTMLフォームでユーザの氏名を要求でき、HTMLラジオ・ボタンを使用して特定の機能を選択するようにユーザに要求できる。これらの要求は、ウェブ・サーバ・アプリケーション222からウェブ・ブラウザ212に送信される。ウェブ・ユーザは、要求された情報を入力し、ウェブ・サーバ・アプリケーション222にそのページの実行要求をもう一度送ることによって応答し、ウェブ・サーバ・アプリケーション222は、ユーザから受信した入力データを解析する。残念ながら、ユーザは、入力の要求に即座に応答しない場合がある。実際、ユーザが、要求された入力を発行する前に数分、数時間または数日待つ可能性があり、その代わりに、トランザクションを完了せずに切断する可能性もあることが予想される。この時間の間、ウェブ・サーバ・アプリケーション222は、通常は、さまざまなプロセスが走行したままの状態、予期される応答を待ちながらその入力待つ。これは、入力を待つ間に走行中のプロセスが活動状態のままにされ、ウェブ・サーバ・アプリケーション222の資源が非効率的に独占されるという問題を引き起こす可能性がある。

【0022】API

アプリケーション・プログラミング・インターフェース（API）は、所与のソフトウェア・アプリケーションの特定の機能へのアクセスを提供するために、プログラム開発者によって使用される。各アプリケーション・プログラムは、第三者が特定の特徴にアクセスでき、そのアプリケーション・プログラムを他のプログラムとインターフェースでき、エンド・ユーザにアクセスを提供できるようにするAPIを有する。各アプリケーション・プログラムは、通常は独自のAPIを有するが、異なるベンダによって提供される異なるAPIの機能性と使用は、かなり似通っていることがしばしばである。多数の異なるソフトウェア・アプリケーションに共通して見られるAPIの例を、図3に示されるコンピュータ・システム300で走行するソフトウェア・アプリケーション342に関して下で提示する。

【0023】SEND APIは、ウェブ・ブラウザ212にデータ（HTMLページか他のタイプの形で）を送信するために発行される。ソフトウェア・アプリケーション342は、SEND APIを呼び出し、SEND APIは、送信要求を送信し、インターネット／アプリケーション・ゲートウェイ（以下ではゲートウェイと呼称する）332とウェブ・サーバ・アプリケーション222を介して適当なウェブ・ブラウザ212にデータを送信する。

【0024】RECEIVE APIは、ウェブ・ブラウザ212からデータを受信するために発行される。これには、必要に応じてCGIデータと環境データを含めることができる。ソフトウェア・アプリケーション34

2は、RECEIVE APIを呼び出して、ゲートウェイ332に入力のを待つように指示する。入力が到着したならば、ゲートウェイ332は、そのデータを処理のためソフトウェア・アプリケーション342に渡す。

【0025】ウェブ・セキュリティ

この1～2年にWWWが爆発的に増大したので、ウェブ・セキュリティに関する関心がますます高まっている。具体的に言うと、ウェブ・ブラウザとウェブ・サーバが、他のコンピュータ資源（すなわち、ソフトウェア・アプリケーション、データ・ファイル、HTMLウェブ・ページなど）へのウェブ・ベースのアクセスを提供するのに使用されている場合に、これらの資源を安全に保たなければならない。これには、これらの資源へのアクセスが、認可されたウェブ・ユーザだけに許可されることを保証することが含まれる。

【0026】場合によっては、異なるユーザが、ウェブ・サーバを介してアクセス可能な異なる資源へのアクセスを許可されるシステムを提供しなければならない。たとえば、あるウェブ・サーバが、2つのソフトウェア・アプリケーションへのウェブ・アクセスを提供し、各ソフトウェア・アプリケーションが、複数のデータベースへのアクセスを提供する場合がある。ユーザの中には、一方のソフトウェア・アプリケーションへのアクセスは必要だが、他方へのアクセスが不要なユーザも、両方のソフトウェア・アプリケーションへのアクセスが必要だが、これらのソフトウェア・アプリケーションによって制御される指定されたデータベースへのアクセスだけが必要なユーザもいる。あるユーザが、2つのソフトウェア・アプリケーションのうちの一方へのアクセスを許可され、セキュリティ検査が、ウェブ・サーバ・レベルだけで行われる場合、そのユーザは、第2のソフトウェア・アプリケーションへのアクセスを許可されていないにもかかわらず、ユーザへのアクセス許可によって、両方のソフトウェア・アプリケーションへのアクセスが許可される。各ソフトウェア・アプリケーションへのアクセスを許可する前にセキュリティ検査が実行される場合であっても、アクセス許可によって、ユーザは、ソフトウェア・アプリケーションによってアクセス可能なデータベースのすべてにアクセスできるようになる。したがって、承認されたユーザのアクセス権を維持しながら、承認されないユーザが重要な資源へのアクセス権を得ることができないことを保証するための、より洗練されたセキュリティ検査技法が必要である。

【0027】通常のウェブ・セキュリティでは、特定のウェブ・サーバへまたはそのウェブ・サーバを介する特定の資源へのアクセスのために特定のウェブ・ユーザを認証するのに、パスワードとユーザIDの組み合わせが使用される。ウェブ・ユーザは、このような保護された資源へのアクセスを試みる時には、ユーザIDとパスワードを供給しなければならない。これは、通常、ウェブ

・ユーザがユーザIDとパスワードを入力することをウェブ・サーバに要求させ、このユーザIDとパスワードをウェブ・ブラウザが記憶し、検証または認証のためウェブ・サーバ・アプリケーションに送り返すことによって達成される。

【0028】この認証処理は、通常は、ウェブ・サーバ・アプリケーションを介してアクセス可能な特定の資源のそれぞれについて繰り返される。したがって、複数の保護資源にアクセスするウェブ・ユーザは、資源ごとにパスワードとユーザIDを入力するよう促され、そうすることが必要になる。たとえば、ユーザは、ウェブ・サーバへのアクセス権を得るために自分のパスワードとユーザIDを入力しなければならない、そのウェブ・サーバを介するソフトウェア・アプリケーションへのアクセス権を得るために第2のパスワードとユーザIDを入力しなければならない、特定のソフトウェア・アプリケーション・データベースへのアクセス権を得るために第3のパスワードとユーザIDを入力しなければならない場合がある。この場合、ユーザは、パスワードとユーザIDを何度も発行する手間に加えて、多数のパスワードとユーザIDを記憶する必要がある。パスワードとユーザIDがどのレベルでも同一の場合であっても、同一の情報を何度も繰り返して入力するのは単調で退屈になる。

【0029】従来のウェブ・サーバ認証システムでは、ウェブ・サーバに実行依頼を送信するたびに、ウェブ・ブラウザがパスワードとユーザIDを再送信する。したがって、パスワードとユーザIDは、「スヌープ」（すなわち、ウェブ・サーバとウェブ・ブラウザの間の伝送の、許可されず望ましくない傍受）の危険に繰り返しさらされる。一部のシステムは、ユーザがパスワードとユーザIDを変更することを定期的に要求することによってこの問題を制限しようとするが、複数のソフトウェア・アプリケーションにアクセスするのに複数のパスワードと複数のユーザIDが必要になる場合には、これが大きな問題になる可能性がある。

【0030】従来技術の解決に関連する問題のために、一部のシステム操作員は、追加のセキュリティ保護手段を除去し、ユーザIDとパスワードを用いるウェブ・サーバ認証だけに頼るようになっている。やはり、この解決を採用できるのは、ウェブからアクセス可能な資源への無許可アクセスの危険性が高くても許容される場合に限られる。

【0031】詳細な説明

本発明による、WWW上でウェブ・ブラウザから複数のソフトウェア・ベンダによって提供される複数のソフトウェア・アプリケーションにグラフィカルな共通ユーザ・インターフェースを提供するための装置及び方法を開示する。このシステムには、ウェブ・ブラウザ、ウェブ・サーバ・アプリケーション、アプリケーション・ゲートウェイ及び少なくとも1つのソフトウェア・アプリケ

ーションを実行する1つまたは複数のコンピュータが含まれる。本発明のシステム及び方法を用いると、ウェブ・ブラウザのユーザが、共通ユーザ・インターフェースを使用して複数のソフトウェア・アプリケーションにアクセスできるようになる。ユーザは、ウェブ・ブラウザを介してデータを入力し、このデータが、ウェブ・サーバ・アプリケーションに通信される。ウェブ・サーバ・アプリケーションは、ウェブ・ブラウザを認証し、ユーザの要求を一意に識別し、追跡するためのデータを含む適当な入力データを、アプリケーション・ゲートウェイに渡す。アプリケーション・ゲートウェイは、ソフトウェア・アプリケーションに対する適当なコマンドを書式化することによって、要求に対する応答を簡単にする。ソフトウェア・アプリケーションは、適当なデータをアプリケーション・ゲートウェイに出力することによって応答する。この出力データには、出力データとその出力データを要求した特定のウェブ・ブラウザの突合せに使用することができる識別子が含まれる。このシステムは、多数のウェブ・ブラウザに、多数のソフトウェア・アプリケーションへの同時アクセスも提供する。さらに、本発明を用いると、標準的なWWW HTMLブラウザのいずれもが、サーバに対する真にプラットフォーム独立のクライアントとして動作できるようになり、すべてのサーバ・サブシステムが対話に用いる一貫した効率的なエンド・ユーザ・インターフェースが提供される。

【0032】理解しやすいGUIインターフェースに統合された時に、本発明の好ましい実施例を構成する主要な構成要素は、MIME (Multipurpose Internet Mail Extension) データ形式を受信し、単一のクライアント・アプリケーションに送信できるWWWトランザクション・サポート技術を提供するカスタム・アプリケーション・プログラミング・インターフェース (API) と、サーバ側セキュリティのためのサポートを提供すると同時に、エンド・ユーザをセキュリティ・システムの複雑さから保護することができるセキュリティ機構と、通常のアプリケーション処理の入出力要件の大半に対処できるHTMLテンプレート/フォームなどの共通WWWユーザ・インターフェース構成要素を定義する機構と、切断、処理タイムアウト、クライアントまたはサーバの異常終了、他の通常の処理の問題などの複雑な問題の処理を自動化する切断処理機構の4つである。この4つのサブシステムのそれぞれを、以下の節で詳細に説明する。

【0033】ここで図1を参照すると、本発明の好ましい実施例によるコンピュータ・システムには、複数の中央処理装置 (CPU) 110、端末インターフェース150、補助記憶装置インターフェース160、ワークステーション170、直接アクセス記憶装置 (DASD) 180、フロッピー・ディスク190、バス140及び、さまざまなソフトウェア・プログラムを格納するための

複数の記憶位置を含むメモリ130が含まれる。この例では、メモリ130に、記憶位置132で走行するウェブ・ブラウザ212、記憶位置134で走行するウェブ・サーバ・アプリケーション222、記憶位置136で走行するゲートウェイ332及び記憶位置138で走行するソフトウェア・アプリケーション342が含まれる。

【0034】CPU110は、システム100の計算機能と制御機能を実行する。システム100に関連するCPUのすべてに、マイクロプロセッサなどの単一の集積回路をそれぞれ個別に含めるか、中央処理装置の機能を達成するため共同して働く適当な個数の集積回路デバイスまたは回路基板を含めることができる。すべてのCPUが、メモリ130内に格納されたプログラムを適当に実行する能力を有し、これらのプログラムまたはシステム100内で発生し得る他の活動に応答して動作する能力を有する。

【0035】メモリ130は、当業者に既知のいずれかのタイプのメモリである。これには、ダイナミック・ランダム・アクセス・メモリ (DRAM)、スタティック RAM (SRAM)、フラッシュ・メモリ、キャッシュ・メモリなどが含まれる。図1には明示されていないが、メモリ130は、単一のタイプのメモリ構成要素とするか、多数の異なるタイプのメモリ構成要素から構成することができる。たとえば、記憶位置132で走行するウェブ・ブラウザ212は、システム100のキャッシュ・メモリの一部とすることができる。さらに、メモリ130とCPU110は、集団としてシステム100を構成する複数の異なるコンピュータの間で分散することができる。たとえば、ウェブ・ブラウザ212が、CPU₁を有するコンピュータに常駐し、ウェブ・サーバ・アプリケーション222が、別のCPU₂を有するもう1つのコンピュータ・システムに常駐し、ゲートウェイ332が、異なるCPU_{n-1}を有する第3のコンピュータ・システムに常駐し、ソフトウェア・アプリケーション342が、異なるCPU_nを有する第4のコンピュータに常駐することができる。図1のシステム100は、CPU110の物理的な位置またはメモリ130内の記憶位置に関する制限なしに、本発明の顕著な特徴の多くを図示するに過ぎない。

【0036】バス140は、システム100のさまざまな構成要素の間でプログラム、データ、状況及び他の形の情報または信号を伝送するために働く。バス140の好ましい実施例は、当業者に既知の、コンピュータ・システム及び構成要素を接続する適当な物理的手段または論理的手段のいずれかである。これには、直接ハードワイヤ接続、インターネット接続、イントラネット接続、光ファイバ、赤外線 (IR) 及び他の形の無線接続が含まれるが、これらに制限されるものではない。コンピュータ・システム及び構成要素を接続するための多数の代

替の方法及び材料を、本発明と共に使用するために簡単に適合できることが予期される。これには、現在既知ではなく、将来に開発される方法及び材料が含まれる。

【0037】端末インターフェース150を用いると、人間のユーザが、通常は、プログラム式であるワークステーション170を介してシステム100と通信できるようになる。図1に示されたシステム100には、単一のワークステーション170だけが含まれるが、実際にシステム100に接続されるワークステーションの数は、システムの設計及びユーザの選択の関数であることを理解されたい。ワークステーション170は、人間がシステム100と対話できるようにする、ダム端末または他の非プログラム式コンピュータ入出力装置とすることもできる。

【0038】補助記憶装置インターフェース160は、当業者に既知の、記憶装置をコンピュータ・システムにインターフェースする方法のいずれかを表す。補助記憶装置インターフェース160を用いると、DASD180などの補助記憶装置を、システム100の他の構成要素に接続できるようになり、これと通信できるようになる。1つの補助記憶装置インターフェース160だけが図示されているが、本発明では、複数のインターフェースとDASD180などの複数の補助記憶装置が予想されている。図1に示されているように、DASD180は、フロッピー・ディスク190上のプログラムまたはデータを読み書きできるフロッピー・ディスク駆動装置とすることができる。DASD180も、当業者に既知の他のいずれかのタイプのDASDとすることができる。これには、CD-ROM駆動装置、ハード・ディスク装置、光ディスク装置などが含まれる。フロッピー・ディスク190は、当業者に既知の通常の3.5インチ磁気媒体ディスクを表す。

【0039】ここで図5を参照すると、本発明の好ましい実施例によるGUIを介してWWW上のソフトウェア・アプリケーションにアクセスするための方法500が示されている。WWWを介するソフトウェア・アプリケーションへのアクセスを開始するために、ユーザは、ウェブ・ブラウザからウェブ・サーバへの入力を生成する何らかの動作を行う（ステップ510）。ウェブ・サーバは、ユーザ要求に関連する必要な動作を実行するためにセキュリティ検査が必要であるかどうかを判定する（ステップ520）。ユーザが、保護資源へのアクセスを要求した場合、適当なセキュリティ検査機構を呼出し、必要な認証を達成する（ステップ522）。ユーザの要求には、さまざまな段階でユーザから入力データを集め、転送することが伴う場合がある。そうである場合には、HTML置換変数を含むHTMLテンプレートを、全体処理またはトランザクションの一部として評価することができる（ステップ530）。必要なHTMLテンプレート及びHTML変数の処理のすべてが、必要

に応じて実行される（ステップ532）。ある時点で、ユーザがウェブ・ブラウザの要求に応答せず、DISCONNECT APIが発行されて、ユーザの要求の処理が一時的に中止される場合がある。その代わりに、前にユーザ対話がすでに発生しており、DISCONNECT APIが以前に呼び出されている可能性もある。どちらの場合でも、現行プロセスを中止するか、前に中止されたプロセスを再開する必要がある（ステップ540）。必要であれば、必要に応じて適当なプロセスを再始動または中止する（ステップ542）。最後に、実際にユーザ要求を評価でき、ソフトウェア・アプリケーションによって適当な応答を生成することができる（ステップ550）。要求された機能を実行した後に、ユーザの要求によって生成された結果が、ウェブ・ブラウザによって表示される（ステップ560）。必要であれば、追加のウェブ・ブラウザ要求を、上に掲げたステップを繰り返すことによって同様の形で処理することができる。

【0040】図5に示された諸ステップを、独立であり、順次であるものとして説明し、図示したが、これらのさまざまなステップは、順次である必要はなく、下のレンタ・カーの例に示されるように、同一のウェブ・トランザクション内で統合化されることが好ましい。これらのステップは、必要に応じて、任意の組み合わせまたは所望の順序で実行できる。一部のソフトウェア・アプリケーションでは、図示のステップのすべてを完了することが必要になる場合があり、他のプロセスや要求では、完了のために図示のステップのうちの1つまたは2つだけが必要になる場合がある。ステップの実際のシーケンスと各ステップの詳細な要件は、主として特定のソフトウェア・アプリケーションの設計選択のままだ。

【0041】セキュリティ機構-FlowMarkの例ここで図2及び図6を参照すると、図5のステップ522は、セキュリティ機構によって実行される。このステップをソフトウェア・アプリケーションのアクセスに関連して説明するが、このステップは、データベースやネットワークなどの他のコンピュータ資源へのアクセスに使用される時にも同等に適用可能である。上で述べたように、クライアント・ワークステーション210のユーザは、ウェブ・サーバ・アプリケーション222にデータを入力することによってWWWを介するソフトウェア・アプリケーションへのアクセス権をウェブ・ブラウザ212に要求させる動作を実行する。入力データには、適当な認証データを入力するのに使用される入力空白を含むHTMLフォームを含めることができる。認証データを入力した後に、ユーザは、このデータをウェブ・サーバ・アプリケーション222に発行する（ステップ621）。好ましい実施例によれば、認証データには、ユーザID、パスワード及びキーが含まれる。ユーザID

及びパスワードには、通常のウェブ・ベースのユーザIDとパスワードを含めることができる。キーには、識別機構として働くデータの組を含めることができる。たとえば、可能なキーの1つが、「flowmarkuser01975」である。

【0042】ウェブ・サーバは、ユーザIDとパスワードを使用して、標準的なウェブ認証手順に従ってユーザを認証する（ステップ623）。ユーザがウェブ・サーバ・アプリケーション222によって正しく認証された時には、ユーザIDとキーが、特定のソフトウェア・アプリケーションへのアクセスの要求と共にゲートウェイ332に渡される（ステップ625）。ゲートウェイ332は、ユーザID及びキーを、ユーザ・ライブラリ620に格納されたユーザID及びキーと比較する（ステップ627）。ユーザ・ライブラリ620には、ゲートウェイ332を介してアクセスされる複数のソフトウェア・アプリケーションのためのユーザ情報が格納されることが好ましい。ユーザ・ライブラリ620は、さまざまな形で構成できる。たとえば、各ソフトウェア・アプリケーションが、認可されたユーザの全員に対して同一のキーを有するようにすることができる。したがって、ゲートウェイ332は、キーを受け取った時に、そのキーによって指定されるアプリケーションに関する、ユーザ・ライブラリ620内の発行されたユーザIDとユーザを比較する。その代わりに、各キーを独自とすることができ、ゲートウェイは、一致するキー及びユーザIDを求めてライブラリ全体を探索することになる。

【0043】具体的な実施態様とは無関係に、ユーザ・ライブラリ620には、許可ユーザごとに、対応するソフトウェア・アプリケーションにアクセスするのに必要な認証データが含まれる。したがって、ゲートウェイ332は、必要なソフトウェア・アプリケーション認証データを取り出すことができる（ステップ629）。このソフトウェア認証データを用いて、ゲートウェイ332は、ソフトウェア・アプリケーション342への通信接続を作成することができる（ステップ631）。具体的に言うと、好ましい実施例では、ユーザ・ライブラリ620から、各ソフトウェア・アプリケーションへの非ウェブ・ベース・アクセス権の認可に通常必要となる認証データと同一の認証データを提供できる。ユーザ・ライブラリ620には、各ソフトウェア・アプリケーションの通常の認証データが格納されるので、ゲートウェイ332は、ソフトウェア・アプリケーション342へのネイティブ・インターフェースによって提供される通常のセキュリティ手順を使用することができる。したがって、ネイティブ・ソフトウェア・アプリケーション・セキュリティ・システムを変更せず、システム管理担当者がソフトウェア・アプリケーション342へのウェブ・アクセスを提供するための特殊なセキュリティ・システムを実施し、維持する必要もなしに、ソフトウェア・ア

プリケーション342への安全なウェブ接続を提供することができる。さらに、ユーザは、ネイティブ・ソフトウェア・アプリケーション・セキュリティ・システムを知る必要も、理解する必要もない。というのは、ユーザはGUIと対話し、このGUIがネイティブ・セキュリティ・システムと対話するからである。

【0044】この例では、キーとユーザIDが、ソフトウェア・アプリケーション342のためにユーザ・ライブラリ620に格納されたユーザID及びキーに対応する。ユーザIDとキーは、ユーザがアクセスを許可されているソフトウェア・アプリケーションの認証情報を突き止めるのに使用される。ユーザIDとキーが、ユーザ・ライブラリ620内で見つからない場合、そのユーザは、要求した資源にゲートウェイ332を介してアクセスすることを許可されない。ユーザ・ライブラリ620に格納されるソフトウェア・アプリケーション認証データには、ワークグループ・ユーザ名、ワークグループ・パスワード、アプリケーション・データベース識別子、または、ソフトウェア・アプリケーション342によって使用されるネイティブ・セキュリティ・システムに従うソフトウェア・アプリケーション342との通信を容易にするのに必要な他の情報を含めることができる。この認証データは、絶対にWWWには送信されないで、許可されないユーザによるスヌープの危険にはさらされない。

【0045】本発明のセキュリティ機構は、ウェブ・ブラウザからWWWを介するソフトウェア・アプリケーションへの安全なアクセス処理及び安全な認証処理を提供する。この機構は、複数のソフトウェア・アプリケーションの認証データを含むユーザのライブラリを使用することによって、ユーザが複数のパスワード及びユーザIDを記憶することを必要とせず、複数のソフトウェア・アプリケーションまたは他のコンピュータ資源への安全なアクセスを提供する。この機構には、ウェブ・ブラウザ、ウェブ・サーバ・アプリケーション及びアプリケーション・ゲートウェイを実行し、適当なセキュリティ検査を実行した後ソフトウェア・アプリケーションを実行する、1つまたは複数のコンピュータが含まれる。この機構を用いると、ウェブ・ブラウザのユーザが、ソフトウェア・アプリケーションにアクセスできるようになる。ユーザは、ウェブ・ブラウザを介してデータを入力し、この入力データが、ウェブ・サーバ・アプリケーションに通信される。ウェブ・サーバ・アプリケーションは、ウェブ・ブラウザを認証し、ウェブ・ブラウザを一意に識別するデータを含む適当な入力データをアプリケーション・ゲートウェイに渡す。アプリケーション・ゲートウェイは、ブラウザから受け取った認証データを使用して、ブラウザのユーザがソフトウェア・アプリケーションのアクセスを許可されるかどうかを判定する。

【0046】好ましい実施例では、ブラウザ認証情報を

使用して、ユーザとソフトウェア・アプリケーションの適当な認証情報とを相関させる。アプリケーション・ゲートウェイは、この新しい認証情報を使用して、ソフトウェア・アプリケーションへのアクセスを得る（すなわちログ・オンする）。

【0047】図4、図6及び図9を参照すると、WWW上でFlowMarkアプリケーション342'にアクセスするのに使用される、本発明のセキュリティ機構が図示されている。上で説明したように、WWW上でFlowMarkアプリケーション342'にアクセスする必要があるユーザは、ウェブ・サーバ・アプリケーション222へのデータ・ストリームを生成する何らかの動作を行う。ウェブ・サーバ・アプリケーション222は、ユーザのデータ・ストリームを調べて、ユーザの要求を満たすために行わなければならない動作を決定する。ユーザが要求した情報が、FlowMarkアプリケーション342'へのアクセスを必要とする場合、ウェブ・ブラウザ212から生成されたデータ・ストリームに、ウェブ・サーバ・アプリケーション222がCGI420に制御を渡すように指令するコマンドが埋め込まれている。この例では、許可されないユーザがWWWを介してFlowMarkアプリケーション342'にアクセスしないようにするために、CGI420へのアクセスが保護されている。CGI420へのアクセス権を得るためには、ユーザは、ウェブ・サーバ・アプリケーション222によって認証されなければならない。ウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212によって生成されたデータ・ストリーム内にCGIコマンドを見つけた時に、ウェブ・ブラウザ212を介してユーザからパスワードとユーザの識別（ユーザID）を要求する。

【0048】さらに、好ましい実施例によれば、ウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212にキーを要求する。ウェブ・ブラウザからウェブ・サーバへキーを提供し、配布する方法には、主に2つの方法がある。まず、キーを、ウェブ・サーバに発行される元々のHTMLページに埋め込むことができる。これは、アクセスの制御が主な問題でない場合に好ましい方法である。たとえば、HTMLフォームにキーを埋め込むことは、発行されるHTMLフォームが、ソフトウェア・アプリケーションの機能のうちの制限された機能にアクセスするために訪問者または「ゲスト」によって使用されるものである場合に有用である。CGIに渡されるフォームにキーを埋め込むことによって、ゲートウェイは、この「ゲスト」キーに対応するソフトウェア・アプリケーションへのアクセスをユーザに許可するように自動的に指令される。さらに、この状況では、ユーザIDとパスワードも埋め込むことができる。ユーザID、パスワード及びキーのすべてがHTMLフォームに埋め込まれていると、ユーザがアクセスを認証されたこ

とを意識することすらなしに、必要な認証が実現される。

【0049】キーは、ゲートウェイがHTMLフォーム内のキーとして認識する値を変数に割り当てることによって埋め込まれることが好ましい。ウェブ・サーバ・アプリケーション222は、フォームを受信し、制御をCGI420に渡し、CGI420は、変数とそれに割り当てられた値を解析し、この情報をFlowMark/インターネット・ゲートウェイ（FMIG）430に渡す。FMIG430は、キーの値及びユーザIDを、ユーザ・ライブラリ620に格納された値と比較する。その後、FMIG430は、キーに基づいて、選択されたソフトウェア・アプリケーションの認証データを取り出す。

【0050】キーを供給するための第2の方法では、ウェブ・サーバ・アプリケーション222が、ユーザがキーを入力することの要求をウェブ・ブラウザ212に送信する。この方法では、ユーザが値を入力する必要があり、したがって、キーに対応するソフトウェア・アプリケーションへのアクセスに追加のセキュリティがもたらされる。通常、ウェブ・サーバ・アプリケーション222からウェブ・ブラウザ212にREALM要求が発行される。REALM要求は、CGI420へのアクセス権を得るためにユーザがユーザIDとパスワードを入力することを要求する、周知のソフトウェア・セキュリティ機能である。

【0051】ウェブ・ブラウザ212がこの要求を受信した時、ユーザは、先に進むために、ユーザID、パスワード及びキーを入力し、これらをウェブ・サーバ・アプリケーション222に発行しなければならない。したがって、ウェブ・ブラウザ212は、ユーザID、パスワード及びキーを記憶し、ウェブ・サーバ・アプリケーション222に送信する。その後、ウェブ・サーバ・アプリケーション222は、ユーザIDとパスワードを使用して、ウェブ・サーバ・アプリケーション222へのアクセス権を有するユーザとしてそのユーザを認証する。ウェブ・サーバ・アプリケーション222は、ユーザから受信したユーザID及びパスワードを記憶し、このユーザIDとキーを、CGI420を介してFMIG430に送る。FMIG430は、ユーザID及びキーを、ユーザ・ライブラリ620の値と比較する。一致が見つかる場合、FMIG430は、そのユーザID及びキーに対応する必要な認証データを、ユーザ・ライブラリから取り出す。

【0052】好ましい実施例では、認証データに、FlowMarkの非ウェブ・ベースのネイティブ・インターフェースを介してFlowMarkにアクセスするのに通常必要となる情報のすべてが含まれる。したがって、ユーザ・ライブラリに格納される認証データには、FlowMarkワークグループ・ユーザ名、Flow

Markワークグループ・パスワード、FlowMarkアプリケーション・データベース識別子、または、発行されたキーに対応するFlowMarkアプリケーションとの通信を簡単にするのに必要な他の情報を含めることができる。この認証データは、絶対にWWWには送信されないで、スヌープの危険にはさらされない。

【0053】FlowMarkと共に使用するのに適した書式を有する、ユーザ・ライブラリ620の一部900を図9に示す。ユーザ（たとえばUser a、User b）ごとに、リストされたユーザのそれぞれがFlowMarkアプリケーション342'にアクセスするのに必要な認証情報を提供する、対応するユーザ名、パスワード、データベース・サーバ識別子及びデータベース識別子がある。ユーザ・ライブラリ620には、FlowMark用の通常の認証データが格納されるので、FMIG430は、FlowMarkに接続する時に通常のセキュリティ手順を使用することができる。したがって、FlowMarkへの安全なウェブ接続を、通常のFlowMarkセキュリティ・システムを変更する必要なしに実現できる。システム管理担当者は、FlowMarkへのウェブ・アクセスのために特殊なセキュリティ・システムを実施し、維持する必要がない。本発明のセキュリティ機構を用いると、FMIG430が、ユーザを認識できるようになり、FlowMarkへの接続を確立できるようになる。

【0054】インターフェース構成要素機構

図2及び図7を参照すると、図5のステップ532は、HTML変数とHTMLテンプレートを使用するインターフェース構成要素機構によって実行される。上で述べたように、ユーザは、ウェブ・サーバ・アプリケーション222にデータを入力することによってWWWを介するソフトウェアへのアクセス権をウェブ・ブラウザに要求させる動作を実行する（ステップ718）。この例では、入力データに、HTMLテンプレートの位置を指定するURLまたは他のアドレス・データが含まれる。ウェブ・サーバ・アプリケーション222は、指定されたテンプレートのURLアドレスをゲートウェイ332に渡す（ステップ720）。ゲートウェイ332は、テンプレート・ライブラリ719内で突き止められた、指定されたHTMLテンプレートを取り出し、オープンする（ステップ721）。テンプレート・ライブラリ719は、ゲートウェイ332と同一の位置に配置されることが好ましいが、ゲートウェイ332によってアクセス可能である限り、どこにでも配置できる。テンプレート・ライブラリ719には、システム開発者がソフトウェア・アプリケーション342のインターフェースを開発する際に例として使用できる、広範囲のHTMLテンプレートが含まれることが好ましい。

【0055】図示の例では、URLアドレスによって、テンプレート・ライブラリ719のテンプレート2が指

定される。その後、ゲートウェイ332が、テンプレート2に格納された変数を識別する（ステップ723）。ゲートウェイ332は、ソフトウェア・アプリケーション342が理解できる適当なコマンドを生成し、そのコマンドをソフトウェア・アプリケーション342に通信する。その後、ゲートウェイ332は、置換変数のそれぞれに必要なデータを、ソフトウェア・アプリケーション342に要求する（ステップ725）。その後、ソフトウェア・アプリケーション342は、入力データの要求を処理する（ステップ727）。入力データの要求を処理する際に、ソフトウェア・アプリケーション342は、データの生成、データを有する他のプログラムの呼出し、または、ローカルまたはネットワーク上の記憶装置からのデータ検索のために、追加のソフトウェア・プロセスを開始することができる。その後、ソフトウェア・アプリケーション342は、要求されたデータをゲートウェイ332に返す（ステップ727）。ゲートウェイ332は、HTMLテンプレート内の変数を、ソフトウェア・アプリケーション342から取り出したデータに置換する（ステップ729）。ゲートウェイ332は、置換変数を実際のデータに置換されたHTMLテンプレートをウェブ・サーバ・アプリケーション222に出力する（ステップ731）。ウェブ・サーバ・アプリケーション222は、ウェブ・サーバ出力データをウェブ・ブラウザ212に供給する（ステップ733）。

【0056】ソフトウェア・アプリケーション342とウェブ・サーバ・アプリケーション222の間のゲートウェイとしてゲートウェイ332を使いやすくするために、HTMLテンプレートのライブラリを設ける。HTMLテンプレートのライブラリによって、複数のCGIモジュールを必要とせず、ソフトウェア・アプリケーション342を含む複数のアプリケーションへのウェブ・ブラウザ212を介するアクセスを提供する、柔軟で簡単にカスタマイズできる方法がもたらされる。したがって、システム操作員は、CGIプログラミング要件を満足する必要なしに、適当なテンプレートを使用することによって、広範囲のアプリケーションに対するカスタム・ウェブ・インターフェースを作成することができる。

【0057】好ましい実施例では、供給されるHTMLテンプレートのそれぞれに、1つまたは複数の変数が含まれる。テンプレートを指定する実行依頼をウェブ・クライアントから受信した時に、ゲートウェイ332がテンプレートをオープンする。ゲートウェイ332は、テンプレートを解析し、テンプレート内の変数を突き止める。その後、これらの変数の値を、それに関連するソフトウェア・アプリケーションに要求する。ソフトウェア・アプリケーション342は、単純に変数に対応するデータを検索することができ、また、適当なデータを生成するためにソフトウェア・プロセスを開始することもで

きる。複数のソフトウェア・アプリケーションに関連する変数を挿入することによって、単一のウェブ・ページによって、これらの複数のソフトウェア・アプリケーションからのデータにアクセスすることができる。データは、ゲートウェイ332によってHTMLテンプレートに解析され、ウェブ・サーバ・アプリケーション222に配布され、ウェブ・サーバ・アプリケーション222は、この出力をウェブ・ブラウザ212に配布する。したがって、正しく定義された変数を有するテンプレートを使用することによって、システム操作員は、すべてのソフトウェア・アプリケーションのために完全に新規のカスタム・インターフェースを作成するという苦勞なしに、ソフトウェア・アプリケーション・データへのウェブ・ベースのアクセスを提供することができる。

【0058】好ましい実施例では、HTMLテンプレートのライブラリに、ソフトウェア・アプリケーションとの対話をウェブ・ブラウザに提供するためにシステム開発者が使用できる広範囲のHTMLページが含まれる。この目的を達成するために、HTMLテンプレートに、使用されるHTMLの複雑さのレベルが異なるさまざまなテンプレートが含まれることが好ましい。たとえば、一部のテンプレートでは、HTMLバージョン3.2でのみ指定されているタグなどの高度なHTMLタグを使用することができる。これらの高度なタグを使用することによって、洗練された書式設定が可能になるが、それに対応した高度なウェブ・ブラウザが必要になる。したがって、これらの高度なタグを含むHTMLテンプレートは、ユーザがソフトウェア・アプリケーションのアクセスに高度なウェブ・ブラウザを使用すると期待するソフトウェア開発者によって使用される可能性がある。逆に、HTML 2.0などのHTMLの以前のバージョンに忠実であり、したがって、広範囲の高度でないウェブ・ブラウザによってアクセスできるHTMLテンプレートも提供できる。広範囲のテンプレートを提供することによって、好ましい実施例では、システム操作員がソフトウェア・アプリケーションへのウェブ・インターフェースを提供する際の高い柔軟性がもたらされる。

【0059】好ましい実施例のHTMLテンプレートには、ウェブ・ブラウザ212とソフトウェア・アプリケーション342の間でデータを渡すのに使用される入力変数が含まれる。さらに、ソフトウェア・アプリケーション342からのカスタマイズされた出力を提供するために、置換変数が使用される。これらの変数は、ゲートウェイ332によって識別できる形でHTMLテンプレートに書き込まれる。このような方法の1つでは、変数をHTMLコメントに書き込み、コメントに配置された変数を解析できるウェブ・サーバを使用する。

【0060】入力変数は、ウェブ・ブラウザ212からゲートウェイ332及びソフトウェア・アプリケーション342へ入力を提供するためにHTMLページに挿入

される。たとえば、次にウェブ・ブラウザ212に送信するHTMLページを指定する入力変数を、HTMLページに含めることができる。この場合、そのHTMLページがウェブ・サーバ・アプリケーション222に送り返された時に、ゲートウェイ332が、変数を解析し、指定されたHTMLページをウェブ・ブラウザ212に配布するようにウェブ・サーバ・アプリケーション222に指示する。

【0061】さらに、ゲートウェイ332は、あるHTML画面から次のHTML画面へ特定の変数を渡すように構成することができる。たとえば、特定の通信インスタンスを表す変数を後続のHTML画面に挿入して、ゲートウェイ332が、特定のユーザのソフトウェア・アプリケーション342との対話を追跡できるようにすることができる。

【0062】置換変数とは、動的データをウェブ・ブラウザに送り返すのに使用される、HTMLテンプレートに含まれる変数である。これらの変数は、ゲートウェイ332によってHTMLテンプレートから解析される。その後、置換変数は、ソフトウェア・アプリケーション342に渡される。ソフトウェア・アプリケーション342は、置換変数によって表される適当なデータを突き止める。このデータは、定義済みのプロセスに従ってソフトウェア・アプリケーション342によって生成することができ、また、単純に適当なデータ記憶装置から取り出すことができる。

【0063】システム操作員は、ユーザ定義変数をHTMLテンプレートに追加して、異なるタイプのソフトウェア・アプリケーションのために必要な特化された入力及び出力を提供することができる。これらの変数は、変数情報が渡されるソフトウェア・アプリケーションまたは変数情報が取り出されるソフトウェア・アプリケーションをゲートウェイ332が決定できる形で命名されなければならない。さらに、HTMLテンプレートに複数の変数を追加することによって、複数の異なるソフトウェア・アプリケーションへの入力を、ウェブ・ブラウザ212からの1つのテンプレートの発行によって行うことができる。同様に、データを複数のソフトウェア・アプリケーション342から取り出し、単一のウェブ・ページに挿入してウェブ・ブラウザ212に出力することができる。

【0064】切断機構

図3及び図8を参照すると、図5のステップ542は、切断機構によって実行される。処理のこの位置では、クライアント・ワークステーション210とソフトウェア・アプリケーション342の間のゲートウェイ332を介する通信がすでに確立されている。これには、一般に、必要であれば、上で説明したウェブ・サーバの認証手段及びセキュリティ手段を使用するユーザの認証が含まれる。さらに、ユーザとソフトウェア・アプリケーション

10

20

30

40

50

ョン342の間でこの「会話」または特定の通信を識別するために、識別子が作成されている。この識別子は、会話識別子と称し、ユーザとソフトウェア・アプリケーション342の間で伝送されるすべての情報に付加され、この特定の通信に属するものとしてその情報にタグを付けるのに使用される。この会話識別子によって、特定のウェブ・ブラウザとソフトウェア・アプリケーション342の間のすべての会話に一意のマークまたはタグが付けられる。会話は、一般に、OPENアプリケーション・プログラム・インターフェース(API)の呼出しによって開始され、一般に、CLOSE APIの呼出しによって終了する。ゲートウェイ332は、会話識別子を使用して個々の要求を記憶する。というのは、ゲートウェイ332が、複数のユーザと、これらのユーザのそれぞれからの複数の要求をサービスする可能性があるからである。したがって、ゲートウェイ332は、さまざまなユーザとソフトウェア・アプリケーション342の間のデータの流れを制御するのに必要な情報を維持する。

【0065】一般に、あるユーザがソフトウェア・アプリケーション342にアクセスした後に、ソフトウェア・アプリケーション342は、そのユーザの要求が完了するか、ウェブ・クライアントからの次の入力が必要になるまで、そのユーザの要求の処理を続ける(ステップ819)。たとえば、次の処理に必要な追加データを突き止めるために適当なデータベースを選択するため、ソフトウェア・アプリケーション342がユーザからの入力が必要とする場合に、処理が停止される可能性がある。この時点で、ソフトウェア・アプリケーション342は、ゲートウェイ332を介してユーザに入力の要求を送信する(ステップ821)。その後、ソフトウェア・アプリケーション342は、ゲートウェイ332に対するDISCONNECT APIを呼び出す(ステップ821)。DISCONNECT APIは、ソフトウェア・アプリケーション342が入力を待っている間にプロセスを中止できるようにするのに使用される切断機構である。DISCONNECT APIは、ゲートウェイ332に、会話識別子を含む、中止される会話に対応する必要なデータ及び状態情報を保管させる。その後、ソフトウェア・アプリケーション342は、現在のソフトウェア・プロセスを中止し、後程ソフトウェア・プロセスを再始動できる状態に戻る。

【0066】その後、ゲートウェイ332は、ウェブ・サーバ・アプリケーションを介して入力の要求をユーザに通信する(ステップ823)。ユーザは、通常、適当なデータをHTMLフィールドに入力することによって、この要求を満たす。ウェブ・クライアントのユーザが、発行ボタンを押した時に、入力されたデータが、ウェブ・サーバとゲートウェイ332を介してソフトウェア・アプリケーション342に送り返される。この応答

は、すぐに返される場合も、比較的長い時間が経過した後に返される場合もある。どの場合でも、ゲートウェイ332は、入力データを送信したウェブ・クライアントを識別し、そのデータが中止されているソフトウェア・アプリケーション342のプロセスのためのものであることを認識する。これは、入力に含まれる会話識別子を、その会話識別子によってインデクシングされる会話のデータベースと突き合わせることによって行われることが好ましい。その後、ゲートウェイ332は、会話識別子に対応する中止されたプロセスを再始動するようにソフトウェア・アプリケーションに指示する(ステップ831)。その後、ソフトウェア・アプリケーション342は、中止されたプロセスを中止された位置から再始動し、入力データを受け取る(ステップ833)。この時点で、プロセスは、必要に応じて、前のように追加処理のためにステップ819に戻るか、終了する。

【0067】好ましい実施例では、DISCONNECT APIを呼び出し、ソフトウェア・アプリケーション342のプロセスを中止するステップが、ウェブ・ブラウザ212からの入力が必要になるたびに行われる。これによって、入力が返されるのを待つプロセスのために必要なコンピュータ資源の量が制限される。DISCONNECT APIは、保守のためにウェブ・サーバ・コンピュータ・システム220、コンピュータ・システム330またはコンピュータ・システム340を一時的に遮断する必要がある時に必ず使用されることが好ましい。これは、走行中のソフトウェア・プロセスを不当に破壊せずに信頼性を確保するために使用することができる。

【0068】ゲートウェイ332には、ウェブ・ブラウザ212とソフトウェア・アプリケーション342の間の対話を簡単にするために設計された複数のAPIが含まれる。このAPIには、OPEN APIとDISCONNECT APIが含まれることが好ましい。さらに、このAPIに、SEND API、RECEIVE API及びCLOSE APIを含めることができる。

【0069】OPEN APIを呼び出すと、ソフトウェア・アプリケーション342とウェブ・ブラウザ212の間の1組の対話が始まる。具体的に言うと、OPEN APIは、呼び出された時に、ソフトウェア・アプリケーション342とウェブ・ブラウザ212の間の会話にタグを付けるのに使用される会話識別子を生成する。会話には、所望のタスクを達成するために複数の異なるプロセスを含めることができる。会話識別子は、個々の通信のそれぞれがどの会話に属するかを識別するために、すべての通信(すなわち、ソフトウェア・アプリケーション342とゲートウェイ332の間の通信と、ウェブ・ブラウザ212とウェブ・サーバ・アプリケーション222の間の通信)に含まれることが好まし

い。

【0070】DISCONNECT APIを用いると、ソフトウェア・アプリケーション342が、必要になるか再始動が可能になるまで、走行中のプロセスを中止できるようにする。これは、ソフトウェア・アプリケーション342のすでに走行しているプロセスを完了するためにウェブ・クライアント・ユーザからの入力が必要な場合に特に有用である。ソフトウェア・アプリケーション342がこのような入力を必要とする時には、ウェブ・サーバ・アプリケーション222を介してウェブ・ブラウザ212に要求を送り返さなければならない。さまざまな理由から、入力が送信されるとしても、その入力がウェブ・サーバ・アプリケーション222を介してソフトウェア・アプリケーション342に送り返されるまでに、数時間もしくは数日を要する可能性がある。DISCONNECT APIを用いると、ソフトウェア・アプリケーション342が、入力を受信するまでそのプロセスを中止できるようにする。具体的に言うと、ソフトウェア・アプリケーションは、そのプロセス（会話識別子によって識別される）を中止することの通知をゲートウェイ332に送り、DISCONNECT APIを呼び出す。DISCONNECT APIは、中止されるプロセスの再始動に必要な情報を保存するようにゲートウェイ332に指示する。したがって、最終的に入力が受信された時に、ゲートウェイ332は、この入力が中止されたプロセス用であることを認識し（関連する会話識別子を使用することによって）、中止されたプロセスを再始動するようにソフトウェア・アプリケーション342に指示し、その入力と必要な状態情報をソフトウェア・アプリケーション342に送り返す。

【0071】したがって、DISCONNECT APIを用いると、ソフトウェア・アプリケーション342とウェブ・ブラウザ212の間の通信の長い遅延が可能になる。DISCONNECT APIを用いなければ、この長い遅延によって、動作を待っている状態のまま未完了のソフトウェア・プロセスがもたらされる可能性がある。DISCONNECT APIは、会話識別子を受け取り、必要な時にゲートウェイ332がプロセスを再起動できるように、必要な情報を保管する。

【0072】CLOSE APIは、特定の会話について、ソフトウェア・アプリケーション342とウェブ・ブラウザ212の間の対話のすべてが完了した時に発行される。ソフトウェア・アプリケーション342が、CLOSE APIを呼び出し、これによって、その会話に完了のタグが付けられ、クローズされた会話に関係する不要なファイル及びデータが除去される。

【0073】SEND APIは、ウェブ・ブラウザ212にデータを（HTMLページまたは他のタイプの形式で）送信するために発行される。ソフトウェア・アプリケーション342が、SEND APIを呼び出し、

これによって、ゲートウェイ332とウェブ・サーバ・アプリケーション222を介してウェブ・ブラウザ212に送信要求が送信され、データが送信される。SEND APIでは、会話識別子を使用して、要求されたデータを必要とするウェブ・ブラウザ212を判定する。

【0074】RECEIVE APIは、ウェブ・ブラウザ212からデータを受信するために発行される。これには、必要に応じてSTDINデータと環境データを含めることができる。ソフトウェア・アプリケーション342は、RECEIVE APIを呼び出して、入力を待つようにゲートウェイ332に指示する。入力が到着した後に、ゲートウェイ332は、会話識別子を使用して入力データを適当な会話と突合せ、必要な処理のためにソフトウェア・アプリケーション342にそのデータを渡す。したがって、ゲートウェイ332は、ソフトウェア・アプリケーション342とウェブ・ブラウザ212の間の通信を容易にするために、OPEN API、DISCONNECT API、CLOSE API、SEND API及びRECEIVE APIを使用する。

【0075】WWWトランザクション・サポート機構図4を参照すると、WWWトランザクション・サポートを説明するために、ウェブ・ブラウザ212からWWWを介して特定のソフトウェア・アプリケーションにアクセスできるようにするシステム400が示されている。この例では、ソフトウェア・アプリケーション342が、特定の機能を実行するようにプログラムされた、FlowMarkと称するワークフロー・アプリケーションである。この特定の例では、FlowMarkを使用して本発明を説明するが、本発明は、FlowMarkに制限されるものではない。他のソフトウェア・アプリケーションを本発明と共に使用することができ、追加のアプリケーション・ソフトウェア・パッケージがそのように使用されることが予期されている。

【0076】FlowMarkは、比較的複雑なプロジェクトまたはタスクを、一連のより小さいプロセスまたはタスクに分解できるようにする、人気のあるプロセス・エンジニアリング・ツールである。FlowMarkによって実行されるアクティビティは、プロセスのフローとアクティビティを記述する1つまたは複数のプロセス・モデル440を使用することによって記述される。これによって、FlowMarkは、通信ネットワークを介してコンピュータ・システムによって達成できる動作及びトランザクションに非常に有用になる。FlowMarkは、一連のコンピュータ化されたシーケンスとして現実のタスクを記述するプロセス・モデル440の作成に使用される。情報は、FlowMarkワークフロー・ソフトウェアによって処理され、通常は、複数の関連アクティビティが伴う。FlowMarkワークフロー・プロセス言語によって、アクティビティがモデル

化され、適当な場合にはトランザクションが自動化される。個々のタスクは、「ワーク・リスト」（すなわち、電子「作業予定」リスト）内に生成される。FlowMarkを使用して特定のタスクを達成する例を、下で詳細に示す。

【0077】図4のシステム400は、図3に示されたコンピュータ・システム300の1つの適当な詳細な実施態様を表す。同一のコンピュータ・システムが存在し（すなわち、クライアント・ワークステーション210、ウェブ・サーバ・コンピュータ・システム220、コンピュータ・システム330及びコンピュータ・システム340）、ウェブ・ブラウザ212がクライアント・ワークステーション210に常駐し、ウェブ・サーバ・アプリケーション222がウェブ・サーバ・コンピュータ・システム220に常駐する。さらに、図3の接続326には、図4の接続426及び428が含まれる。図4のシステム400は、FlowMarkアプリケーション342'との通信に使用される時のゲートウェイ332の具体的な実施例を示すために本明細書に提示されたものである。

【0078】この特定の実施例について、ゲートウェイ332には、CGI420、FMIG430及びWWWアプリケーション・プログラム・インターフェース（API）434が含まれる。4つの標準的なWWW APIが、Open、Close、Send及びReceiveである。WWW API434によって、FMIG430とアクティビティ・プログラム432の間の通信リンクがもたらされる。FlowMarkアプリケーション342'には、FlowMark450と、1つまたは複数のアクティビティ・プログラム432が含まれる。FlowMark450には、FlowMarkデータベース438、1つまたは複数のプロセス・モデル440及びFlowMark API436が含まれる。特定のソフトウェア・アプリケーション342は、FlowMark内でプロセス・モデル440を定義し、プロセス・モデル440内の特定のタスクを実行するアクティビティ・プログラム432を定義することによって、システム400内で実施される。FlowMark API436は、FlowMark450と共に供給される標準APIであり、これによって、アクティビティ・プログラム432とFMIG430がFlowMark450と通信する手段がもたらされる。FlowMarkデータベース438は、プロセス・モデルに関する情報を格納するのに使用することができる、汎用データベースである。たとえば、プロセス・モデル440とアクティビティ・プログラム432によって、レンタ・カー・ワーク・フロー・プロセスが実施される場合、FlowMarkデータベース438は、どの自動車を使用可能であるかなど、レンタ・カー・プロセスに関する情報の格納に使用されるはずである。

【0079】WWWを介してFlowMarkアプリケーション342'にアクセスする必要があるユーザは、クライアント・ワークステーション210を使用してウェブ・ブラウザ212に要求を入力する。ユーザは、特定のホーム・ページ・サイトのURLを入力するか、ウェブ・ブラウザ212を使用する、HTMLによって生成されたユーザ・インターフェースに提示されるボタンをクリックすることができる。ユーザが、通常はHTMLフォームに表示されたボタンのクリックによって、必要な情報を「発行」する時に、ウェブ・サーバ・アプリケーション222が、ウェブ・ブラウザ212から入力データを受信する。このデータ・ストリームは、RFC1866形式やRFC1867形式を含む多数の異なる形式でウェブ・サーバ・アプリケーション222に提示できる。この2つの具体的な形式は、一般的なウェブ・ブラウザが理解する共通データ・ストリーム形式の例にすぎない。本発明は、これらの形式に制限されるものではなく、現在既知であるか将来に開発されるデータ伝送形式のすべてを含む。

【0080】ユーザの要求した情報のために、FlowMarkアプリケーション342'へのアクセスが必要な場合、CGI420へのアクセスの必要を識別するコマンドがデータ・ストリームに埋め込まれており、このCGI420が、FlowMarkアプリケーション342'へのアクセスを提供する。この特定の例では、このコマンドがCGI420の呼び出しである。CGI420の呼出しは、URLの一部とすることができ、その代わりに、ウェブ・ブラウザ212によって発行されるデータに埋め込むことができる。CGI420へのアクセスは、許可されないユーザがWWWを介してFlowMarkアプリケーション342'にアクセスしないようにするために、保護されている。CGI420へのアクセス権を得るには、ユーザは、ウェブ・サーバ・アプリケーション222によって認承されなければならない。図6に関して上で述べた認証機構が、好ましい認証機構である。ウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212によって生成されたデータ・ストリーム内でCGIコマンドを見つけた時に、ウェブ・ブラウザ212を介してユーザにパスワードとユーザ識別（ユーザID）を要求する。ユーザが認証された後に、ウェブ・サーバ・アプリケーション222は、CGI420に制御を渡して、必要な動作を実行する。

【0081】CGIは、当業者に周知の実行可能プログラムである。CGIは、WWWを介して情報を送信するための機構を提供するように機能する。CGI420は、ウェブ・サーバ・アプリケーション222からFMIG430へ、コマンドの形で情報を渡す。このコマンドは、一方向すなわちCGI420からFMIG430に向かうが、データと状況は、両方向で受け渡しされる。ウェブ・サーバ・アプリケーション222は、CG

1420を呼び出し、要求に関する適当なコマンドとデータを送る。本発明のこの好ましい実施例では、ウェブ・ブラウザ212からCGI420へのデータ伝送に「CGI Post」(stdin)形式を使用するが、ウェブ・ブラウザ212によって生成できる他のすべてのデータ伝送形式が予期され、本発明の範囲に含まれる。さらに、CGI420の解析機能及び他の動作機能は、他の形で実施できることに留意されたい。たとえば、多数のウェブ・サーバ・アプリケーションが、現在「モジュール」をサポートしている。モジュールとは、動的リンク・ライブラリ(DLL)を使用することによって実施されるソフトウェア・ルーチンである。モジュールは、図4のシステム400内でCGI420と同一の機能を実行することができ、本発明の好ましい実施例のいくつかでは、CGI420と置換できる。モジュール及びDLLの使用は、当業者に周知である。したがって、CGI420の使用は、例示のみを目的とし、本発明の制限ではない。

【0082】ウェブ・ブラウザ212からデータを受信した後に、CGI420は、データを解析して、FlowMarkアプリケーション342'へのアクセスの要求を含む、要求された処理に関する関連情報を突き止める。CGI420は、ユーザのデータ及び要求を、いくつかの制御情報と共にFMIG430に送る。FMIG430は、FlowMarkアプリケーション342'がWWWを介してウェブ・ユーザと対話する方法を提供する。FMIG430は、CGI420とFlowMarkアプリケーション342'の間の情報の流れを指示し、FlowMark API436を使用することによってFlowMark機能を開始する。たとえば、FMIG430は、FlowMark API436を呼び出して、ユーザによって発行された要求を処理するのに必要なプロセス・インスタンスを作成することができる。その後、FMIG430は、異なるFlowMark API436を使用して、このプロセス・インスタンスの呼出しまたは起動を行うことができる。このプロセスは、プロセス・モデル440によって支配され、プロセス・モデル440は、所望のタスクを実行するために呼び出さなければならないアクティビティ・プログラム432をFlowMark450に知らせる。FMIG430は、FlowMarkプロセスを起動した後に、要求が処理されたことを示す情報を、FlowMark API436を介してFlowMark450から受け取るか、WWW API434を介してアクティビティ・プログラム432から受け取るまで待機する。FMIG430とFlowMark API436の間のコマンド・インターフェースは、一方向すなわち、必ずFMIG430がFlowMark API436を呼び出すが、データと状況情報は、両方向に流れる。これによって、ウェブ・インターフェースが変更された場合で

あっても、FlowMarkアプリケーション342'へのインターフェースを無変更のままにすることができるので、これは重要である。

【0083】また、FMIG430は、FlowMarkとの対話を必要とする各ウェブ・クライアントからの要求に、会話識別子を割り当てる。上で説明したように、会話識別子によって、特定のウェブ・ブラウザとFlowMarkの間のすべての会話に一意のマークまたはタグが付けられる。会話は、一般にOPENアプリケーション・プログラム・インターフェース(API)を呼び出すことによって開始され、一般にCLOSE APIを呼び出すことによって終了する。FMIG430は、複数のユーザとそれらのユーザのそれぞれからの複数の要求をサービスする可能性があるため、会話識別子を使用して個々の要求を記憶する。したがって、FMIG430は、さまざまなユーザとFlowMarkによって処理されているプロセス・インスタンスとの間の情報の流れを制御するのに必要な情報を維持する。

【0084】アクティビティ・プログラム432は、ユーザが要求したタスクを達成するためにFlowMarkが使用できるソフトウェア・モジュールである。個々のアクティビティ・プログラム432は、FlowMarkによって開始され、その後、そのアクティビティ・プログラム432は、WWW API434を介してウェブ・クライアントと通信する。各ワークフローのプロセス・モデル440によって、所望のタスクを達成するのに必要なアクティビティ・プログラム432が開始される。各アクティビティ・プログラム432は、所望のタスクを達成するために走行し、要求された情報を返し、その後終了するプログラムのインスタンスである。たとえば、アクティビティ・プログラム432は、ユーザの要求を処理するためにユーザに情報を要求する場合がある。アクティビティ・プログラム432は、適当なWWW API434を呼び出して、必要なデータを得る。この場合、FMIG430は、アクティビティ・プログラム432への発行を待っているデータを有し、このデータは、この処理でCGI420が前に送ったデータと同一である。アクティビティ・プログラム432は、WWW API434を呼び出して、データの要求をFMIG430に送り、FMIG430は、WWW API434からの要求に応答して、このデータをアクティビティ・プログラム432に返す。アクティビティ・プログラム432は、このデータを処理するために適当なステップを実行し、要求を満たす。アクティビティ・プログラム432は、WWW API434、FMIG430、CGI420、ウェブ・サーバ・アプリケーション222及びウェブ・ブラウザ212を介してユーザと通信する。また、アクティビティ・プログラム432は、WWW API434から戻りコードを受け取って、ユーザが要求したデータを受信したことを検証す

る。確認を受信した後に、アクティビティ・プログラム432のこの特定のインスタンスは、その要求を完了し、終了する。1つまたは複数のアクティビティ・プログラム432の他のインスタンスが、まだ活動状態であり、他の要求を処理している可能性がある。

【0085】WWW API434は、FMIG430とアクティビティ・プログラム432の間の対話を提供するために働く。WWW API434を用いると、アクティビティ・プログラム432が、ウェブ・クライアントとの間でデータ及び状況を送受信できるようになる。FlowMark API436が、無変更のままであることを留意することが重要である。これが重要であるのは、ウェブ・ユーザがFlowMarkにアクセスできるようにするためにFlowMark APIが変更されることがないからである。この意味では、FlowMarkは、ウェブ・クライアントからアクセスされていることを知らず、専用アプリケーションが要求された機能を実行している場合と同一の形で動作する。FlowMark APIは影響を受けないので、FlowMarkの動作は、FlowMarkアプリケーション342'が達成することをウェブ・クライアントが要求しているプロセスまたはタスクに無関係に、変更されない。複数の位置からの複数のユーザが、WWWを介してFlowMarkにアクセスでき、FlowMarkインターフェースが変更されないことが保証される。場合によっては、システムが透過的なウェブ・クライアントを提示するが、これは、ユーザの要求がFlowMarkアプリケーション342'によって処理されていることがユーザに示されないことを意味する。

【0086】ウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212と同一の位置に配置できることに留意されたい。さらに、必要ではないが、コンピュータ・システム330とコンピュータ・システム340は、好ましい実施例では同一のコンピュータ・システムである。最後に、図面に示された接続は、当技術分野で既知の、コンピュータ・システムを接続するためのあらゆるタイプの物理的手段または論理的手段とすることができる。これには、直接接続、インターネット接続、イントラネット接続、赤外線（IR）及び他の形態の無線接続が含まれるが、これらに制限されるものではない。コンピュータ・システムを接続するための多数の代替の方法及び材料を、本発明と共に使用するために簡単に適合できることが予期される。要するに、本明細書では複数の別々のコンピュータ・システム（たとえば図3及び図4）が開示されるが、本発明によるコンピュータ・システムには、あらゆる台数のコンピュータ・システムまたはコンピュータ・システムの組み合わせを含めることができる。本明細書の図面は、本発明の顕著な態様を例示するために示されたものであって、本発明を本明細書に示された特定の構成に制限するものと解釈しては

ならない。

【0087】共通ユーザ・インターフェースを介する複数のソフトウェア・アプリケーションへの複数ユーザのアクセス

図2と図10を参照すると、本発明のもう1つの長所は、複数のユーザがWWW上でウェブ・ブラウザ212を介して同時に複数のソフトウェア・アプリケーションにアクセスする時に最も明らかになる。各ウェブ・ユーザは、一般に、WWWへのアクセスに使用されるクライアント・ワークステーション210を有する。図10の特定の構成では、クライアント・ワークステーション1（CW1）及びクライアント・ワークステーション2（CW2）が、ウェブ・サーバ1（WS1）に結合され、クライアント・ワークステーション3（CW3）が、ウェブ・サーバ2（WS2）に結合される。クライアント・ワークステーションCW4及びCW5は、ウェブ・サーバ3（WS3）に結合される。ウェブ・サーバWS1及びWS2の両方が、ゲートウェイ・コンピュータ・システム（GCS）、この場合にはGCS1に結合される。図からわかるように、WS3は、GCS2に結合される。GCS1とGCS2は、どちらもがソフトウェア・アプリケーション・コンピュータ1（SAC1）に結合される。さらに、GCS2は、ソフトウェア・アプリケーション・コンピュータ2（SAC2）にも結合される。SAC1とSAC2の両方で、ウェブ・ユーザがWWWを介するアクセスを必要とする可能性があるさまざまなソフトウェア・アプリケーションが走行している。

【0088】本発明は、ウェブ・クライアントのそれぞれに共通グラフィカル・ユーザ・インターフェースを提供し、SAC1及びSAC2で走行するアプリケーション・ソフトウェア・プログラムへのすべてのクライアント・ワークステーション（CW1ないしCW5）によるアクセスのすべてを、要求された機能が要求通りに実行されることが保証される形で管理する。GCSは、ウェブ・クライアントとソフトウェア・アプリケーションの間でトラフィックの方向を指定する、アプリケーション・ゲートウェイを提供する。本発明のセキュリティ機構は、アプリケーション・ソフトウェアと共に動作し、さまざまなコンピュータ資源のそれぞれの認証データへのアクセスに使用されるユーザIDとキーを必要とする。会話のそれぞれに一意の識別子を割り当てることによって、GCSは、SAC上で走行するソフトウェア・アプリケーションのそれぞれと、そのネイティブ・インターフェースを使用して対話することができ、その後、ソフトウェア・アプリケーションからの出力を識別し、その出力を、出力を要求したウェブ・クライアントと突き合わせることができる。本発明によるHTMLテンプレートとHTML変数を使用することによって、一部のウェブ・クライアントは、異なるコンピュータ・システム上

の複数のソフトウェア・アプリケーションからの情報に実際にアクセスすることができ、要求したデータを単一のウェブ・ページ内で受信することができる。どの時点でも、ウェブ・クライアントのうちの1つが、処理を中止され、ソフトウェア・アプリケーションへの入力供給を停止する可能性がある。本発明のDISCONNECT API機能を使用することによって、コンピュータの時間が無用に浪費されなくなる。

【0089】自動車レンタルの例

本発明の好ましい実施例の具体的な例を、ある人が、WWWを介してレンタ・カー代理人のFlowMarkアプリケーション・ソフトウェアにアクセスすることによって自動車を借りたいと思っている状況を使用することによって、詳細に説明できる。図26を参照すると、自動車レンタルの例全体のプロセス・モデル2000には、プロセス・ステップ2010、2020、2030、2040及び2050が含まれる。プロセス・モデル2000のプロセス・ステップ2010では、アクティビティ・プログラムが自動車レンタル情報を受け取り、次の予約番号を突き止め、その予約番号をファイルに保管し、要求者に予約番号を返し、FlowMark出力データ・コンテナに予約番号をセットする。プロセス・ステップ2010では、FlowMarkのInternet Connection WWW API及びデータ・コンテナAPIを使用する。プロセス・ステップ2020は、予約を満たすために使用可能な自動車があるかどうかを判定するアクティビティ・プログラムである。プロセス・モデル2000の後続経路のどちらに進むかは、この判定の結果に依存する。予約を満たすために使用可能な自動車がある場合、プロセス・モデル2000の次のステップは、プロセス・ステップ2030になる。しかし、予約要求を満たすために使用可能な自動車がない場合には、プロセス・モデルの次のステップは、プロセス・ステップ2040になる。プロセス・ステップ2030では、自動車レンタル要求に一致する、使用可能な自動車が識別され、予約要求を満たすためにスケジューリングされる。さらに、予約確認番号が生成され、要求者に転送される。プロセス・ステップ2040では、レンタ・カー代理人は、予約要求を満たすために別の位置からレンタ・カーを捜す試みを行うか否かを選択できる。別の位置にある使用可能なレンタ・カーを捜す場合、プロセス・モデル2000の次のステップは、プロセス・ステップ2050になる。プロセス・ステップ2050では、予約要求を満たすために自動車をスケジューリングし、ある位置から別の位置へ移動するためのアクティビティ・プログラムを使用できる。プロセス・ステップ2050では、別の位置で使用可能な自動車のリストが提示され、その結果、レンタ・カー代理人が、移動するのに適した自動車を選択できるようにする。これらのステップのそれぞれを、以下で詳細に説

明する。

【0090】図4と図26を参照すると、自動車を借りたい人またはユーザは、ウェブ・ブラウザ212が走行しているクライアント・ワークステーション210を使用することによってWWWにアクセスする。ユーザは、ウェブ・ブラウザ212を使用して、レンタ・カー代理店のURLを入力し、レンタ・カー代理店のホーム・ページ・サイトを突き止める。その後、ユーザは、レンタ・カー代理店のウェブ・サイトで、自動車レンタルの要求を受け入れるように以前に設定された適当な区域またはページを突き止める。この例では、レンタ・カー代理店のウェブ・サイトにレンタル予約フォームがあると仮定する。自動車レンタル予約フォームを突き止めた後に、ユーザは、レンタ・カー代理店が要求する情報を入力する。この情報には、通常、顧客識別番号、パスワード、自動車を借りる都市、レンタ・カー借用希望日、借りる自動車の具体的な車種（すなわち、メーカー、モデル、サイズ）などの項目が含まれる。

【0091】適当なレンタル予約フォームまたはテンプレート生成するHTMLの例を、図11に示す。ウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212に送り返す必要がある次の出力として図11のHTMLテンプレートを指定する入力をウェブ・ブラウザ212から受信すると仮定する。この指定は、ウェブ・ブラウザ212からのフォーム入力の使用を含む、さまざまな方法で行うことができる。このフォームでは、ユーザが、会員番号、姓、名、出発点の都市、州、開始日、日数及び自動車の好みを入力することを要求される。ユーザは、この情報を入力した後に、レンタル予約フォーム上の「発行」ボタンをクリックすることによって、この情報を発行する。この時点で、ウェブ・サーバ・アプリケーション222が、ユーザ要求によって生成されたデータ・ストリームをウェブ・ブラウザ212から受信する。適当な書式によって、すべての変数及び他の関連情報データを拾い上げ、図12のデータ・ストリームのような外見のpostデータ・ストリーム形式でウェブ・サーバ・アプリケーション222に送る。上で説明したように、このデータ・ストリームは、多数の異なるデータ形式でウェブ・サーバ・アプリケーション222に提示することができ、本発明は特定のデータ形式には制限されない。この動作は、図26のプロセス・ステップ2010に含まれる。

【0092】ウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212からのデータ・ストリームを検査して、ユーザの要求を満たすために行わなければならない動作を決定する。この特定の例では、HTMLフォームの「<FORM ACTION="/cgi-prot/exmp5cgi.exe" METHOD="POST">」という行のCGI420への呼出しから、ウェブ・サーバ・アプリケーション222は、exmp5cgi.exeと称する保護CGI（CGI420と仮定す

る)を呼び出す必要があることを知る。ウェブ・サーバ・アプリケーション222は、この保護CGIへの呼出しを見た時に、ウェブ・サーバ・アプリケーション222がCGI420へのアクセスを許可する前に認証が必要であることを知る。

【0093】CGI420は保護されているので、許可されないユーザがWWWを介してFlowMarkアプリケーション342'にアクセスできなくするために、CGI420へのアクセスを制御しなければならない。CGI420を指定したHTMLフォーム内のパラメータに回答して、ウェブ・サーバ・アプリケーション222は、ウェブ・ユーザを認証しなければならない。この認証では、ユーザがユーザIDとパスワードを入力する必要があり、ウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212にREALM要求を発行する。REALM要求は、CGI420へのアクセス権を得るためにユーザがユーザIDとパスワードを入力することを要求する、周知のソフトウェア・セキュリティ機能である。ウェブ・ブラウザ212は、ユーザから受け取ったユーザIDとパスワードを記憶し、この情報をウェブ・サーバ・アプリケーション222に送信する。ウェブ・サーバ・アプリケーション222は、このユーザIDとパスワードを使用してユーザを認証してから、そのユーザにCGI420へのアクセスを許可する。ウェブ・ブラウザ212の将来の実施態様のいくつかでは、ウェブ・サーバ・アプリケーション222に送信されるデータ・ストリームにウェブ・ブラウザ212が何らかの形の認証データを挿入する機構が設けられる可能性が高いことに留意されたい。認証データを収集し、ウェブ・ブラウザ212からウェブ・サーバ・アプリケーション222に送信するための方法及び技法のすべてが、本発明の範囲に含まれる。

【0094】認証の後に、ウェブ・サーバ・アプリケーション222は、CGI420に制御を渡し、CGI420は、ウェブ・ブラウザ212からデータを受信する。この特定の例では、CGI420は、標準入力プロトコル「stdin」を介して「post」形式のデータ・ストリームを受信する。post形式は、当技術分野で周知のデータ形式記述である。stdinは、postデータを読み取るファイル記述子である。本発明のこの好ましい実施例では、postデータ形式記述が使用されるが、ウェブ・ブラウザ212及びウェブ・サーバ・アプリケーション222によって使用できる他のすべてのデータ形式が予期され、本発明の範囲に含まれる。この例でウェブ・ブラウザ212からCGI420に送信されるレンタル予約フォームのpostデータを、図12に示す。

【0095】CGI420は、ウェブ・ブラウザ212からpostデータを受信した後に、データを解析し、ワークフロー変数を探す。この例では、ワークフロー変

数は接頭辞wf-を有する。2つの隠されたワークフロー変数が、「wf-cgi-submit=2」及び「wf-cgi-html=\exam\smp\exmp5srk.html」である。最初の変数「wf-cgi-submit=2」は、CGI420に、FlowMarkプロセス・インスタンスを作成し、始動する必要があることを知らせる。2つの隠されていないワークフロー変数が、「wf-fmig-key」と「wf-api-proc-template='WWW.Reservation.Request」である。これらの変数は、作成し、始動しなければならないプロセス・テンプレートを示す。単一のHTMLテンプレートで複数の変数を使用して、複数のアプリケーションからデータを取り出すことができる。たとえば、3つの変数を含むHTMLテンプレートを指定できる。ある変数で、始動しなければならない特定のFlowMarkプロセスを指定でき、他の2つの変数を使用して、FlowMarkアクティビティ・プログラムによってアクセス可能な異なるソフトウェア・アプリケーションから情報を取り出す。HTMLテンプレートがウェブ・サーバ・アプリケーション222に発行される際に、CGIは、変数を解析し、FMIGに配布する。FMIGは、wf変数によって指定されたプロセスを始動するようソフトウェア・アプリケーションに指示し、2つのアプリケーション変数を適当なソフトウェア・アプリケーションに渡す。ソフトウェア・アプリケーションは、変数に対応するアプリケーションから適当なデータを取り出す。したがって、適当なHTML変数を使用することによって、ウェブ・クライアントは、単一のウェブ・ページを介して多数の異なるアプリケーションを用いて作業することができる。この対話は、ウェブ・クライアントにとって完全に透過的とすることができ、また、インターフェース・ウェブ・ページから明白になるようにすることができる。

【0096】したがって、置換変数を有するHTMLテンプレートを使用することによって、単一の比較的単純なCGIモジュールが、FMIGとあいまって、ウェブ・サーバと複数のソフトウェア・アプリケーションの間の効果的なインターフェースを提供することができる。これによって、システム操作員は、複数のCGIモジュールを使用する必要も、極度に複雑なCGIモジュールを使用する必要もなしに、WWWを介する複数のソフトウェア・アプリケーションへの簡単にカスタマイズできるウェブ・アクセスを提供できるようになる。CGI420は、他の制御情報と共に、postデータと環境データをFMIG430に送る。環境データは、当業者に周知の標準プログラミング技法を介して入手可能であり、標準「C」プログラミング言語パラメータとしてCGI420に渡すことができる。

【0097】FMIG430は、FlowMark API436を使用してCGI420からの要求を通信することによって、CGI420とFlowMarkアプリケーション342'の間の情報の流れを指示する。こ

これらのFlowMark APIは、FlowMark製品と共に出荷される標準APIであり、本発明に適合するために変更する必要はない。これによって、新しいウェブ・インターフェースを設ける場合であっても、FlowMarkアプリケーション342'のネイティブ・コマンド・インターフェースを無変更のままにすることができるので、これは重要である。FMIG430とFlowMarkアプリケーション342'の間のコマンド・インターフェースは一方向すなわち、必ずFMIG430がFlowMark API436を呼び出す10が、データと状況情報は両方向に流れることに留意されたい。

【0098】CGI420からデータを受け取った後に、FMIG430は、データを解析し、wf-fmig-key変数からユーザID及びキーを突き止めて、対応するFlowMarkユーザIDと、そのユーザにFlowMarkアプリケーション342'へのアクセスを提供する他のFlowMarkアクセス情報があるかどうかを判定する。ユーザIDとキーに対応するFlowMarkアクセス情報があると仮定すると、FMIG43020は、このアクセス情報を使用してFlowMarkアプリケーション342'にログ・インする。FMIG430は、FlowMark API436のうちの1つを呼び出して、ユーザによって発行された要求を処理するのに必要なプロセス・インスタンスを作成する。この例では、要求によって、「www.Reservation_Request」と称するプロセス・インスタンスが作成される。このプロセス・インスタンスは、WWW自動車レンタル・トランザクションを処理するように特別に設計されている。その後、別のFlowMarkAPI436を使用して、30FMIG430が、このプロセス・インスタンスを呼び出すか始動する。また、FMIG430は、「wf-cgi.html」変数用の適当なHTML変数情報を保管する。「wf-cgi.html」は、ワークフロー・シーケンスが終了した後に変更すべきHTMLをFMIG430に知らせる変数である。FlowMarkプロセスwww.Reservation_Requestの接頭辞によって、このプロセスがウェブ使用可能すなわち、アクティビティ・プログラム432がWWWAPI434を使用してFMIG430と通信することが、FMIG430に知られる。FMIG430が、プロセス・インスタンスとそれを要求したウェブ・ブラウザを対応させることができることを確実にするために、FMIG430は、このウェブ・クライアントの「ハンドル」を生成し、記憶する。プロセスに応じて、このハンドルは、プロセス・インスタンス名、アクティビティ・インスタンス名及びウェブ・クライアントのセキュリティ・データのなんらかの組み合わせになる。このハンドルは、ウェブ・クライアントの一意の識別子である。その後、FMIG430は、ハンドル、プロセス・インスタンス、ユーザIDなどを、その内部データ・40

キャッシュに入力する。その後、FMIG430は、WWWAPI434の接続を待つ。この時点で、CGI420は、まだ接続316を介してFMIG430に接続されており、ユーザの自動車レンタル要求を満たすために、要求された状況またはデータをFMIG430から受け取るのを待っている。

【0099】自動車レンタル予約プロセス・モデルが初めて作成された時に、第1のアクティビティ・プログラム432が、識別され、FlowMark自動車レンタル予約プロセス・モデルが呼び出された時に必ず自動的に実行されるように指定された。これは、FMIG430が自動車レンタル予約プロセス・モデルのFlowMarkインスタンスを作成し、起動する時に、必ず、第1の自動車レンタル予約FlowMarkアクティビティ・プログラムであるアクティビティ・プログラム432が、FlowMarkアプリケーション342'によって自動的に始動されることを意味する。この特定の例では、自動車レンタル要求を処理するために一緒に働く複数の関連するアクティビティ・プログラム432がある。FlowMarkワークフロー・プロセス言語によって、自動車レンタル要求及び予約プロセスがモデル化され、適当な場合にトランザクションが自動化される。自動車レンタル要求を処理するために、FlowMarkアプリケーション342'によって、FlowMarkデータベース438内に自動車レンタル予約プロセス・モデルのインスタンスが作成され、プログラム実行クライアント（PEC）が活動状態であるかどうかを検査される。PECは、FlowMarkアプリケーション342'内のアクティビティの動作と流れを調整する、監視プログラム・モジュールである。PECが活動状態になった後に、FlowMarkアプリケーション342'は、PECにアクティビティをディスパッチし、FlowMarkデータベース438を更新して、アクティビティが走行中であることを示す。その後、PECは、このプロセスの第1のアクティビティに登録されたアクティビティ・プログラム432を始動する。その後、選択されたアクティビティ・プログラム432が走行を開始する。ウェブ・クライアントの自動車レンタル要求を処理するのに必要なアクティビティ・プログラム432の数と性質は、プロセス・モデル440をどのように設計したかに依存する。たとえば、あるプロセス・モデル440では、アクティビティが、完了まで人間の介入なしに走行する完全に自動的なプロセスとしてモデル化される可能性がある。その代わりに、プロセス・モデル440で、モデル・プロセスを完了する前に広範囲の人間の入力及び介入が必要になる場合もある。設計の如何にかかわらず、各ワークフローのプロセス・モデル440によって、モデル化されたプロセスまたは手順を完了するために指定されたタスクを達成するのに必要な特定のアクティビティ・プログラム432が開始され

る。

【0100】個々のアクティビティ・プログラム432のそれぞれは、特定のタスクを達成するか、要求された情報を返し、その後終了するように設計された、別々のソフトウェア・モジュールである。WWW API434を用いると、アクティビティ・プログラム432が、WWWを介してウェブ・クライアントとの間でデータ及び状況を送受信できるようになる。WWW API434は、必ずアクティビティ・プログラム432によって呼び出されるが、データ及び状況情報は、FMIG430とアクティビティ・プログラム432の間で両方向に流れる。

【0101】この例のレンタ・カー要求を処理するために、複数の関連するアクティビティ・プログラム432を呼び出して、プロセス・モデル440で指定されたワークフロー・プロセスを完了することができる。たとえば、1つのアクティビティ・プログラムを開始して、FlowMarkデータベース438に問い合わせ、レンタ・カーを要求している人物がそのレンタ・カー代理店の既存の顧客であるかどうかを判定することができる。そうである場合には、顧客識別番号が、FlowMarkデータベース438内の既存のアカウント番号と一致し、アクティビティ・プログラム432は、次に進んで自動車レンタル要求を処理できる。しかし、レンタ・カーを要求している人物が既存の顧客でない場合、異なるアクティビティ・プログラム432を開始して、ウェブ・クライアントと通信し、必要な情報を集め、ウェブ・クライアントの顧客情報をFlowMarkデータベース438に入力することができる。

【0102】次に、有効な顧客識別が確立された後に、もう1つのアクティビティ・プログラム432を開始し、顧客の要求に従って要求を満足できるかどうかを判定することができる。これは、図26のプロセス・ステップ2020に含まれる。たとえば、要求された日に、要求された都市に、要求されたサイズの使用可能な自動車があるかどうかを判定する。その代わりに、FlowMarkプロセス・モデルで、次の処理のために自動車レンタル要求を人間の代理人に経路指定することを指定することもできる。この場合、自動車レンタル要求は、代理人のFlowMarkタスク・リストに表示される。その代わりに、このプロセス全体を完全に自動化することもできる。どの場合でも、レンタ・カー代理人またはアクティビティ・プログラム432は、ウェブ・クライアントの自動車レンタル要求を処理し、所望の自動車を使用可能な場合には、FlowMarkデータベース438を更新して、その自動車が「予約」済みであることを示す。これらのアクティビティは、図26のプロセス・ステップ2030で達成される。しかし、所望の自動車を使用可能でない場合には、別のアクティビティ・プログラム432を開始して、別の位置で自動車を探

し(図26のプロセス・ステップ2040)、人間のレンタ・カー代理人から、所望の自動車を別の位置から所望の位置へ移動する要求を作成することができる(図26のプロセス・ステップ2050)。これらのアクティビティ・プログラム432の一部を、下で詳細に説明する。

【0103】この例では、レンタ・カー代理人が、要求を是認し、車両を所望の位置に移動した後に、レンタ・カー代理人は、FlowMarkアプリケーション342'に入力を与え、アクティビティ・プログラム432を更新する。レンタ・カーがこのシステムで予約された後に、異なるアクティビティ・プログラム432を開始して、その顧客の確認番号を生成し、その顧客との確認のトランザクションを処理することができる。各アクティビティ・プログラム432は、最後まで実行した後に終了する、独立のプロセスとして設計される。

【0104】この時点で、アクティビティ・プログラム432は、FlowMark API436を使用して、FlowMark入力データ・コンテナから入力データを取り出す。FlowMarkデータ・コンテナは、プロセス・モデル440の作成時に定義されるFlowMark機能である。FlowMarkデータ・コンテナは、FlowMark API436を介してアクセスでき、プロセス・モデル440内で状況及び情報を記憶し、あるアクティビティ・プログラム432から次のアクティビティ・プログラム432へ渡すための記憶位置として使用される。アクティビティ・プログラム432は、接続428を介してWWW API434にOpenを発行することによって、ウェブ・クライアントとの会話をオープンする。アクティビティ・プログラム432には、実行中のプロセス・インスタンス、アクティビティ名、FlowMarkユーザIDなどの情報も含まれる。この情報は、WWW API434によってFMIG430に送られる。FMIG430は、内部データ・キャッシュにある要求されたプロセス・インスタンス名を突き止めることによって、このOpen APIを適当なウェブ・クライアントと突き合わせる。その後、FMIG430は、このトランザクションの「会話識別子」を生成する。FMIG430は、その内部キャッシュに会話識別子を保管し、応答メッセージの一部としてWWW API434に会話識別子を送る。WWW API434は、FMIG430からの切離しを行い、会話識別子をアクティビティ・プログラム432に返す。この時点から、この会話識別子は、ウェブ・クライアントの要求を処理するのに必要な、このウェブ・クライアントとすべてのアクティビティ・プログラム432の間のWWW API434発行のすべてに含まれる。この会話識別子は、アクティビティ・プログラム432が発行するClose APIによって会話を終了するまで有効である。

【0105】FMIG430から会話識別子を受け取った後に、アクティビティ・プログラム432は、WWW API434にReceiveを発行する。Receive APIは、FMIG430に対する、ウェブ・クライアントからデータを取得する要求である。この場合、FMIG430は、アクティビティ・プログラム432への発行を待っている、ウェブ・クライアントからのデータを有する。これは、このプロセスで前にCGI420がFMIG430に送ったものと同一のウェブ・クライアント・データである。FMIG430は、このpostデータ及び環境データをWWW API434 (Receive)に送り、Receive APIによって、そのデータがアクティビティ・プログラム432に中継される。アクティビティ・プログラム432は、そのデータを処理し、要求を満たすために適当なステップを実行する。

【0106】このプロセスのこの時点で、アクティビティ・プログラム432は、クライアント・ワークステーション210のウェブ・クライアントに予約確認HTML画面を送信するSendをWWW API434に発行する。このSend APIによって、FMIG430に対する送信要求が生成され、予約確認画面を描画するのに必要なHTMLデータが送られる。FMIG430は、要求に含まれる会話識別子とウェブ・クライアント・ハンドル（一緒に内部データ・キャッシュに記憶されている）を突合せ、まだFMIG430に接続されて入力待っているCGI420に、データ型及びハンドルと共にアクティビティ・プログラム432からのHTMLデータを送る。アクティビティ・プログラム432は、データ型「HTMLテンプレート」を生成し、その結果、CGI420が、元のHTMLテンプレートを解析でき、アクティビティ・プログラム432によって送られたHTMLデータを用いて適当なHTML置換変数を置換できるようにする。その代わりに、アクティビティ・プログラム432は、MIME、URLまたはHTMLテンプレート・データをCGI420に送ることができる。この時点で、CGI420は、FMIG430からの切離しを行い、受け取ったデータの処理を開始する。その後、FMIG430は、応答メッセージ内のOKの戻りコードによってWWW API434に応答する。WWW API434も、FMIG430からの切離しを行い、アクティビティ・プログラム432への戻りコードを生成する。一般に、WWW API434は、各WWW API434が呼び出された後にFMIG430からの切離しを行う。

【0107】予約確認テンプレートのHTMLコードの例を、図13に示す。変数wf-act-outmsgは、ウェブ・ブラウザ212によって表示されるHTML画面内の確認番号によって置換される置換変数であることに留意されたい。FMIG430から受け取ったデータをCGI

420が処理した後のHTMLコードを、図14に示す。図14のHTMLコードは、ウェブ・サーバ・アプリケーション222が変更するコードであり、ウェブ・ブラウザ212がクライアント・ワークステーション210に表示するコードである。図14のHTMLコードには、顧客の予約番号が含まれることに留意されたい。この場合、CGI420とウェブ・クライアントは、すでに会話を確立しており、CGI420は、初めて呼び出された時からまだ活動状態であるから、ウェブ・クライアントを認証する必要はない。CGI420は、隠された変数「wf-fmig-handle」を確認番号と共に挿入し、また、最初のHTML画面からのwf-fmig-keyをこのHTMLコードに挿入する。これによって、後続のHTMLページに、そのページからFlowMark450へのアクセスを許可する埋込みデータを含めることができるようになる。この例に示されたHTML変数の置換は、JavaScript変数の置換及び複製と、JavaScriptテンプレートの解析及び挿入を含むように拡張することができる。本発明は、HTML変数の置換に制限されない。

【0108】予約確認テンプレートの処理中に、アクティビティ・プログラム432は、WWW API434にReceive APIを発行しており、これが、顧客が確認番号を受信し、見たことを保証する、アクティビティ・プログラム432のための確認メッセージとして働く。ウェブ・クライアントは、自分の確認番号を受信し、記録した時に、「発行」ボタンをクリックする。このデータは、前と同様に、ウェブ・ブラウザ212によって保持されているウェブ・ユーザのユーザID及びパスワードと共に、ウェブ・サーバ・アプリケーション222に送信される。ウェブ・サーバ・アプリケーション222は、この情報を使用して、もう一度CGI420に対してウェブ・クライアントを認証する。ウェブ・サーバ・アプリケーション222は、前と同一のHTMLコード、<FORM ACTION="/cgi-prot/exmp5cgi.exe" METHOD="POST">を用いてCGI420を呼び出す。CGI420は、ウェブ・ブラウザ212からのpostデータを受け取る。この例では、ウェブ・ブラウザ212は、データを「form-urlencoded media type」として書式化する。このデータは、図15に示されたものに類似した外見になる。13というwf-cgi-submitの値は、データをアクティビティ・プログラムに渡すコマンドである。FMIG430は、wf-fmig-keyを検索し、postデータからのキーと環境データからのウェブ・ユーザIDを検査して、ウェブ・クライアントが許可されることを確認する。FMIG430は、wf-fmig-handleを内部データ・キャッシュの会話識別子と突合せ、WWW API434が接続されているかどうかを検査する。WWW API434がまだ接続されていない場合、FMIG430は単に待機する。CGI420は、FMIG

430に接続されたままになり、応答を待ち続ける。ReceiveのWWW API434がFMIG430に接続された後に、FMIG430は、会話識別子を使用して、Receive APIを適当なウェブ・クライアントと突合せ、postデータと環境データをWWW API434に送り、WWW API434は、このデータをアクティビティ・プログラム432に渡す。アクティビティ・プログラム432は、データを処理し、Close WWW APIを使用して会話をクローズする。Close APIは、会話識別子、プロセス・インスタンス名及び他の関連データと共にFMIG430に送られる。FMIG430は、クローズ要求を処理し、応答メッセージをWWW API434に送り、WWW API434は、会話が終了したことを検査するため、アクティビティ・プログラム432への戻りコードを生成する。

【0109】その後、FMIG430は、内部データ・キャッシュから記憶された変数値を取り出し、そのデータをCGI420に送る。CGI420は、データと変数を受け取り、FMIG430からの切離しを行う。その後、CGI420は、ウェブ・サーバ・アプリケーション222に連絡し、変数とデータを送り、ウェブ・ブラウザ212への指定されたHTMLを描画するようにウェブ・サーバ・アプリケーションに指示する。

【0110】この時点で、自動車レンタル要求に関するユーザとの対話は完了している。しかし、その間に完了している可能性がある、上で述べた他のアクティビティ・プログラム432がある。さらに、他のプロセスが、完了を必要とする可能性があり、これらのプロセスを達成するために他のアクティビティ・プログラム432が活動化される可能性がある。この例では、1つのアクティビティ・プログラム432が、ユーザと対話し、ユーザから集めたデータをFlowMarkデータ・コンテナに送っていた。FlowMarkデータ・コンテナのデータは、現在は、プロセス・モデル440内の別のアクティビティ・プログラム432のための入力になっている。他のアクティビティ・プログラム432の一部を、下で説明する。

【0111】アクティビティ・プログラム432のいずれかが動作を完了した後に、制御がPECに戻される。PECは、FlowMarkにリターンし、FlowMarkデータベース438が、1つのアクティビティ・プログラム432が完了したことと、このプロセスを完了するための次のアクティビティ・プログラム432を開始できることを示すように更新される。次のアクティビティ・プログラム432が、自動的に実行されるアクティビティ・プログラム432である場合、FlowMarkアプリケーション342'は、FlowMarkデータベース438内にアクティビティのインスタンスを作成し、PECが活動状態であるかどうかを検査し、

アクティビティをPECにディスパッチし、アクティビティが走行中であることを示すようにFlowMarkデータベース438を更新する。PECは、次に走行するアクティビティとして登録されているアクティビティを始動し、適当なアクティビティ・プログラム432が、完了するまで走行する。

【0112】上で注記したように、自動車レンタル要求を処理するのに必要なアクティビティの一部を、人間の対話または介入を必要とするようにモデル化することができる。たとえば、必要な場合には、自動車の予約要求を検査するために、人間のレンタ・カー代理人を必要とすることができる。その代わりに、要求された自動車が要求された位置で使用可能でない場合に、別の位置から適当な自動車を移動するために人間の代理人に連絡することができる。ウェブ・クライアントの自動車の要求の結果は、レンタ・カー代理店が要求された日に宛先の都市で使用可能な要求された自動車を有するか否かの2つがありえる。FlowMarkデータベースへの照会によって使用可能性を検査した後に、正しいアクティビティ・プログラムが開始される。

【0113】この例では、アクティビティ・プログラム432が、FlowMarkユーザのワーク・リストの項目を生成し、FlowMarkユーザ、この場合ではレンタ・カー代理人が、アクティビティを手動で始動しなければならない。レンタ・カー代理人は、ウェブ・ブラウザ212を使用して、FlowMark実行時クライアント・ウェブ・ページにアクセスする。レンタ・カー代理人用の適当なウェブ・ページを生成するのに必要なHTMLコードは、図16に示されたコードに類似したものになるはずである。レンタ・カー代理人は、自分のキーを入力し、「Work with Work Items (ワーク項目に関する作業)」を選択し、発行ボタンをクリックする。このラジオボタンは、図16のHTMLに<INPUT TYPE="radio" NAME="wf-cqi-html" VALUE="/exm/html/exmp5ewi.htm" CHECKED>という行があるので、特定のHTMLテンプレートすなわちexmp5ewi.htmに関連する。

【0114】やはり、前に述べたように、ウェブ・サーバ・アプリケーション222は、ウェブ・ブラウザ212から生成されたpostデータ・ストリームを受信する。CGI420は保護されているので、もう1つのREALM要求を発行し、応答を得なければならない。前と同様に、このREALM要求では、レンタ・カー代理人がパスワードとユーザIDを入力することが必要である。ウェブ・ブラウザ212の将来の実施態様のいくつかでは、ウェブ・サーバ・アプリケーション222に送信されるデータ・ストリームにウェブ・ブラウザ212が何らかの形の認証データを挿入する機構が設けられる可能性が高いことに留意されたい。認証データを収集し、ウェブ・ブラウザ212からウェブ・サーバ・アプリケーション222に送信するための方法及び技法のす

べてが、本発明の範囲に含まれる。一般に、各ウェブ実行時クライアントは、ウェブ上でのユーザIDとパスワードを知っていなければならない。どの場合でも、ウェブ・ブラウザ212は、ウェブのユーザIDとパスワードを記憶し、この情報をウェブ・サーバ・アプリケーション222に送信する。前と同様に、ウェブ・サーバ・アプリケーション222は、ウェブ・ユーザID及びパスワードを使用して、ウェブ・クライアントを認証し、CGI420へのアクセス権を得る。レンタ・カー代理人が認証された後に、CGI420が、上の例のHTML言語、`<FORM ACTION="/cgi-prot/exmp5cgi.exe" METHOD="POST">`を介して呼び出される。CGI420は、`stdin`を介してウェブ・サーバ・アプリケーション222から`post`データ・ストリームを受け取る。このデータ・ストリームは、通常は図17の例に似た外見になる。

【0115】データ・ストリームの発行(`wf-cgisubmit`)の値が0なので、CGI420は、指定されたテンプレート(`exmp5ewi.htm`)を取り出し、HTML変数についてテンプレートを解析し、解析された変数を、環境データ及び制御情報と共にFMIG430に渡す。CGI420が処理を開始する前にテンプレートを生成するのに使用されるHTMLコードの例は、図18及び図19に類似するはずである。

【0116】FMIG430は、データ・ストリームからレンタ・カー代理人のキー値(`webfmagent`)とウェブ・ユーザIDを取り出し、この情報を使用して、レンタ・カー代理人が使用を許可されているFlowMarkユーザIDを、FlowMarkデータベース438から見つける。レンタ・カー代理人は、まだFlowMark450にログ・インしていないので、FMIG430は、FlowMarkデータベース438内の相関するFlowMark情報を使用してFlowMark450にログ・インする。FMIG430は、FlowMark API436を使用して、ユーザによって前に入力された、自動車レンタル要求に関連する情報(すなわち、`wf-api-item-id`、`wf-api-item-descrip`、`wf-api-item-state`など)を取り出し、まだFMIG430に接続され、FMIG430からのデータを待っているCGI420のためのメッセージにこのデータを書式化する。このデータは、FMIG430を介してCGI420に渡される。データがCGI420に送られた後に、CGI420は、FMIG430からの切離しを行い、受け取ったデータの処理を開始する。CGI420は、FMIG430からの情報を取り、変数を処理し、その情報を、クライアント・ワークステーション210に表示される適当なテンプレートに置く。CGI420が処理を完了した後のHTMLコードの例を、図20、図21及び図22に示す。図20、図21及び図22に示されたHTMLコードは、クライアント・ワークステーション

ョン210上でレンタ・カー代理人用のワーク・リストのワーク項目の出力画面を描画するのに使用される。レンタ・カー代理人は、FlowMark450によって生成され、ウェブ・ブラウザ212によって表示されたワーク項目のリストを見る。レンタ・カー代理人は、表示されたワーク項目のうちの1つを選択し、「start work item (ワーク項目開始)」ボタンをクリックし、「perform action (アクション実行)」ボタンをクリックする。レンタ・カー代理人によるこれらの動作によって、ウェブ・ユーザID及びパスワードの情報を含む、ウェブ・サーバ・アプリケーション222へのデータ・ストリームが生成される。ウェブ・ユーザID及びパスワードは、ウェブ・サーバ・アプリケーション222によって保持され、レンタ・カー代理人がこの情報を再入力する必要はない。ウェブ・サーバ・アプリケーション222は、記憶したユーザID及びパスワードを使用して、レンタ・カー代理人を認証し、CGI420へのアクセス権を得る。ウェブ・サーバ・アプリケーション222は、前に使用したものに類似したHTMLコード、`<FORM ACTION="/cgi-prot/exmp5cgi.exe" METHOD="POST">`を用いてCGI420を呼び出し、CGI420は、ウェブ・ブラウザ212から`post`データを受信する。`post`データは、`stdin`を介してCGI420によって受信されるが、図23に示されたデータに類似する。

【0117】この場合、レンタ・カー代理人がワーク項目開始の要求を発行しているので、`submit`変数の値は3に等しい。CGI420は、指定されたテンプレート(この場合は`exmp5ewi.htm`)を検索し、HTML変数についてテンプレートを解析し、これらの変数を、データ・ストリーム、環境データ及び制御情報と共にFMIG430に渡す。この時点で制御はFMIG430に渡されるが、CGI420は、FMIG430に接続されたままで、FMIG430からデータが返されるのを待つ。FMIG430は、CGI420が送った変数を使用して、レンタ・カー代理人を認証する。レンタ・カー代理人は、すでにFlowMark450にログ・オンしているので、FMIG430は、レンタ・カー代理人をもう一度ログ・オンする必要はない。FMIG430は、データ・ストリームを解析し、ワーク項目(`wf-api-item`)開始の要求を見つけ、ワーク項目の処理を開始するためにFlowMark API436を発行する。ワーク項目は、HTML変数を用いてWWWコンテキストで送信されているので、FMIG430は、アクティビティ・プログラムがWWW API434を使用してウェブ・クライアントとの会話を有することを知っている。FMIG430は、このウェブ・クライアントのハンドルを生成し、ハンドル、プロセス・インスタンス名、アクティビティ名、ユーザIDなどを内部キャッシュに入力する。また、FMIG430は、HTML変

数情報を保管し、その結果、会話が行われる時にHTML変数を復元できるようにする。その後、FMIG430は、WWW API434の接続を待つ。

【0118】要求されたワーク項目(wf-api-item)を開始するFlowMark API436に回答して、FlowMarkは、FlowMarkデータベース438内にアクティビティのインスタンスを作成し、PECが活動状態であるかどうかを検査し、アクティビティをPECにディスパッチし、FlowMarkデータベース438を更新してアクティビティが走行中であることを示す。PECは、このプロセスで次のアクティビティに登録されているアクティビティ・プログラム432を始動する。アクティビティ・プログラムは、完了まで走行する。

【0119】アクティビティ・プログラム432は、FlowMark APIを使用して、FlowMark入力データ・コンテナから入力データを取り出す。アクティビティ・プログラムは、WWW API434にOpen APIを発行することによってウェブ・クライアントとの会話をオープンする。Open APIによって、プロセス・インスタンス、アクティビティ名、FlowMarkユーザIDなどの情報を含む、FMIG430用のOpen要求メッセージが生成される。FMIG430は、この情報を受け取り、内部データ・キャッシュ内の適当なプロセス・インスタンス、アクティビティ名及びユーザIDを見つけることによって、ウェブ・クライアントを照合する。前と同様に、FMIG430は、会話識別子も生成し、この識別子を内部キャッシュに記憶する。FMIG430は、応答メッセージでWWW API434に会話識別子を送り、WWW API434は、FMIG430からの切離しを行い、会話識別子をアクティビティ・プログラム432に返す。この時点から、会話識別子が、このウェブ・クライアント及びこのプロセス・インスタンスを扱うすべてのWWW

API434の発行に含まれるようになる。会話識別子は、この会話がクローズされるまで有効になる。アクティビティ・プログラム432は、WWW API434にSendAPIを発行して、ウェブ・クライアントへのHTML画面を生成する。アクティビティ・プログラム432は、データ型としてHTMLテンプレートを指定し、その結果、CGI420に、テンプレートを解析し、アクティビティ・プログラム432によって送られたデータを用いて変数を置換することを知らせる。アクティビティ・プログラム432は、テンプレートの位置と、各変数の置換のためのテキストを指定する。

【0120】Send APIによって、FMIG430に対する送信要求が生成され、データが送信される。FMIG430は、送信要求に含まれる会話識別子と適当なウェブ・クライアントを突き合せ、アクティビティ・プログラム432からのデータ、データ型及びハンドル

を、まだ接続されているCGI420に送る。この時点で、CGI420は、FMIG430からの切離しを行い、受け取ったデータの処理を開始する。FMIG430は、WWW API434に回答し、WWWAPI434も、FMIG430からの切離しを実行する。WWW API434は、アクティビティ・プログラム432に戻りコードを送る。CGI420は、FMIG430から送られたデータを使用して、顧客の要求を満たすために使用可能な自動車に関する情報を含む画面を作成する。さらに、CGI420は、ウェブ・サーバ・アプリケーション222と通信しているので、CGI420は、隠された変数「wf-fmig-handle」と、最初のHTML画面からのwf-fmigキーを、新しい画面のHTMLコードに挿入する。その後、CGI420は、テンプレートを解析し、「wf-」変数を処理して、適当な位置に情報を書き込む。テンプレート・ファイルのHTMLコードは、図24に示されたHTMLコードに類似した外見になる。CGI420が処理を終了した後は、ファイル内のHTMLコードは、図25のHTMLコードに類似した外見になる。これは、ウェブ・サーバ・アプリケーション222が、クライアント・ワークステーション210上のレンタ・カー代理人の画面を描画するのに使用されるHTMLコードである。その間に、アクティビティ・プログラム432は、ReceiveのWWWAPI434を発行して、ウェブ・クライアントからデータを受信している。レンタ・カー代理人は、この自動車レンタル要求を満たすために予約する自動車を選択し、「submit process (プロセス発行)」ボタンをクリックする。ウェブ・ブラウザ212、ウェブ・サーバ・アプリケーション222、CGI420、FMIG430及びFlowMark450の間の通信処理は、上の説明と同様に行われる。最終的に、選択された自動車が、FlowMarkデータベース438内で更新され、会話が終了する。自動車レンタル予約プロセス・インスタンスは、この時点で完了するので、FlowMark450は、FlowMarkデータベース438からこのプロセス・インスタンスを除去する。

【0121】同様のイベントのシーケンスが、要求された位置に使用可能なレンタ・カーがない状況でも発生する。唯一の相違は、この場合、要求された都市に条件を満たす自動車がないので、レンタ・カー代理人の画面に、使用可能な自動車に関する情報が含まれないことである。この画面には、代替位置にある、顧客の要求に一致する自動車を移動するオプションが含まれるはずである。

【0122】その間に、アクティビティ・プログラム432は、Disconnect APIを発行している。Disconnect APIによって、会話識別子、プロセス・インスタンス、アクティビティ名、FlowMarkユーザIDなどの情報を含む、FMIG4

10

20

30

40

50

30に関する切断要求メッセージが生成される。FMIG430は、この情報を受け取り、内部キャッシュ内の対応する項目を突き止め、この情報のすべてを、アクティビティ・プログラム432の「切断」状況と共に記録する。FMIG430は、WWW API434にOKの応答メッセージを送り、WWW API434は、FMIG430からの切離しを行い、アクティビティ・プログラム432に戻りコードを送る。アクティビティ・プログラム432は、戻りコードを検査し、切断APIが成功裡に記録されたことを検査する。アクティビティ・プログラム432は、切断されたプロセスを後程復元するために、会話識別子と状況情報を局所データベースに保管する。このアクティビティは、実際に完了することなくできる限り進行しているので、制御はPECに戻る。アクティビティは完了していないので、PECは、FlowMarkデータベース438を更新して、アクティビティが再始動の準備ができていることを示す。FlowMarkに関しては、これは「手動始動」アクティビティである。このアクティビティが、作動可能であり、手動始動であるので、FMIG430は、後程、レンタ・カー代理人からのデータが使用可能になった時に、FlowMark API436を発行して、アクティビティを再始動することができる。レンタ・カー代理人は、自動車レンタル要求を満たすために代替位置から自動車を移動することを選択し、「発行」をクリックする。データ転送とウェブ・クライアント認証の処理は、前に説明した形で進行する。この場合、FMIG430は、会話識別子を検査した時に、アクティビティの状況が「切断」であることを知る。FMIG430は、FlowMark API436を発行して、アクティビティ・プログラム432を再始動する。したがって、アクティビティ・プログラム432が再始動され、前と同様に、FlowMark 450がプロセス・インスタンスを作成し、FlowMarkデータベース438を更新する。PECが、アクティビティ・プログラム432を始動し、必要なデータが、FMIG430から転送される。この時点で、アクティビティ・プログラム432は、完了まで走行することができる。どのレンタ・カーをどの位置から移動するかを判定し、類似したシナリオ及び並びの通信によって、このデータをレンタ・カー代理人からFlowMark 450へ送ることが可能である。

【0123】上に示した例からわかるように、WWW上でさまざまなソフトウェア・アプリケーションにアクセスするための共通ユーザ・インターフェースを提供することによって、高い効率と共に高い生産性が可能になる。ユーザがさまざまなソフトウェア・アプリケーションへのアクセス権を得るのに必要な訓練の長さや量の両方を減らしつつ、より多くのソフトウェア・アプリケーションを使用するための柔軟性も得られる。これらの長

所のすべてが、競争力があり、市場で成功するビジネスを可能にする。

【0124】本発明の好ましい実施例に関して本発明を具体的に図示し、説明してきたが、当業者であれば、本発明の趣旨及び範囲から逸脱することなく、形態及び詳細におけるさまざまな変更を行うことができることを理解するであろう。

【0125】まとめとして、本発明の構成に関して以下の事項を開示する。

【0126】(1) 少なくとも1つの中央処理装置(CPU)と、CPUに結合されたメモリと、メモリ内に常駐し、少なくとも1つのCPUによって実行され、共通ユーザ・インターフェースを介して複数のウェブ・ブラウザへまたはこれらからデータを送受する能力を有し、データの識別及び追跡のために識別子機構を使用する、トランザクション・サポート機構とを含む、ワールド・ワイド・ウェブ上で複数のウェブ・ブラウザとソフトウェア・アプリケーションとの間で通信するための共通ユーザ・インターフェースを提供するコンピュータ・システム。

(2) さらに、セキュリティ機構を含み、セキュリティ機構が、メモリ内に常駐し、少なくとも1つのCPUによって実行され、セキュリティ機構が、ソフトウェア・アプリケーションと複数のウェブ・ブラウザとの間に結合され、ソフトウェア・アプリケーションと複数のウェブ・ブラウザとの間のインターフェースを提供し、セキュリティ機構が、複数のウェブ・ブラウザからユーザ入力を受け取り、セキュリティ機構が、受け取った入力に対応するソフトウェア・アプリケーションの認証パラメータを取り出す、上記(1)のコンピュータ・システム。

(3) さらに、インターフェース機構を含み、インターフェース機構が、少なくとも1つの変数を処理するためのゲートウェイ機構を含み、ゲートウェイ機構が、メモリ内に常駐し、少なくとも1つのCPUによって実行され、ゲートウェイ機構が、ソフトウェア・アプリケーションの再プログラミングを必要としない、複数のウェブ・ブラウザとソフトウェア・アプリケーションとの間の通信のための汎用共通ゲートウェイ・インターフェースを含む、上記(1)のコンピュータ・システム。

(4) さらに、切断機構を含み、切断機構が、メモリ内に常駐し、少なくとも1つのCPUによって実行され、切断機構が、ソフトウェア・アプリケーション・プロセスが再開される時にデータを取り出すことができるように、ソフトウェア・アプリケーション・プロセスが中止される時に、複数のウェブ・ブラウザのうちの1つとソフトウェア・アプリケーション・プロセスとの間の会話のそれぞれに関連する状態データ及び会話識別子を記憶する、上記(1)のコンピュータ・システム。

(5) メモリ内に常駐し、少なくとも1つのCPUによ

って実行され、ソフトウェア・アプリケーションと複数のウェブ・ブラウザとの間に結合され、ソフトウェア・アプリケーションと複数のウェブ・ブラウザとの間のインターフェースを提供し、複数のウェブ・ブラウザからユーザ入力を受け取り、受け取った入力に対応するソフトウェア・アプリケーションの認証パラメータを取り出す、セキュリティ機構と、少なくとも1つの変数を処理するためのゲートウェイ機構を含み、ゲートウェイ機構が、メモリ内に常駐し、少なくとも1つのCPUによって実行され、ゲートウェイ機構が、ソフトウェア・アプリケーションの再プログラミングを必要としない、複数のウェブ・ブラウザとソフトウェア・アプリケーションとの間の通信のための汎用共通ゲートウェイ・インターフェースを含む、インターフェース機構と、メモリ内に常駐し、少なくとも1つのCPUによって実行され、ソフトウェア・アプリケーション・プロセスが再開される時にデータを取り出すことができるように、ソフトウェア・アプリケーション・プロセスが中止される時に、複数のウェブ・ブラウザのうちの1つとソフトウェア・アプリケーション・プロセスとの間の会話のそれぞれに関連する状態データ及び会話識別子を記憶する、切断機構とをさらに含む、上記(1)のコンピュータ・システム。

(6) トランザクション・サポート機構が、さらに、ソフトウェア・アプリケーションへのネイティブ・インターフェースと通信するための機構を含む、上記(1)のコンピュータ・システム。

(7) トランザクション・サポート機構が、ウェブ・サーバ・アプリケーション及びソフトウェア・アプリケーションと通信するアプリケーション・ゲートウェイを含み、アプリケーション・ゲートウェイが、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行され、アプリケーション・ゲートウェイが、識別子機構を含み、識別子機構が、複数のウェブ・ブラウザのそれぞれのために識別子を生成し、ソフトウェア・アプリケーションからのデータを、複数のウェブ・ブラウザのうちの識別子に対応する選択された1つに経路指定する、上記(1)のコンピュータ・システム。

(8) アプリケーション・ゲートウェイが、複数のウェブ・サーバから受け取るデータを処理し、アプリケーション・プログラムから受け取るデータを処理する、上記(7)のコンピュータ・システム。

(9) ソフトウェア・アプリケーションが、プロセス・エンジニアリング・ソフトウェア・アプリケーションである、上記(1)のコンピュータ・システム。

(10) さらに、ソフトウェア・アプリケーションの指示の下で実行される少なくとも1つのアクティビティ・プログラムと通信する少なくとも1つのアクティビティ・プログラム・インターフェース(API)を含み、少なくとも1つのアクティビティ・プログラム・インター

フェースが、少なくとも1つのアクティビティ・プログラムとアプリケーション・ゲートウェイとの間で通信する、上記(1)のコンピュータ・システム。

(11) 複数の中央処理装置(CPU)と、複数のCPUに結合されたメモリと、それぞれがメモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、複数のウェブ・ブラウザと、複数のウェブ・ブラウザのうちの少なくとも1つと通信する、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、ウェブ・サーバ・アプリケーションと、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、ソフトウェア・アプリケーションと、ウェブ・サーバ・アプリケーション及びソフトウェア・アプリケーションへのネイティブ・インターフェースと通信する、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、アプリケーション・ゲートウェイとを含み、アプリケーション・ゲートウェイが、複数のウェブ・ブラウザのそれぞれのために識別子を生成し、ソフトウェア・アプリケーションからのデータを、複数のウェブ・ブラウザのうちの識別子に対応する選択された1つに経路指定する、識別子機構を含むワールド・ワイド・ウェブ上でウェブ・ブラウザとソフトウェア・アプリケーションとの間で通信するための共通ユーザ・インターフェースを提供するコンピュータ・システム。

(12) アプリケーション・ゲートウェイが、ウェブ・サーバ・アプリケーション及びアプリケーション・プログラムから受け取るデータを処理する、上記(11)のコンピュータ・システム。

(13) ソフトウェア・アプリケーションが、プロセス・エンジニアリング・ソフトウェア・アプリケーションである、上記(11)のコンピュータ・システム。

(14) さらに、ソフトウェア・アプリケーションの指示の下で実行される少なくとも1つのアクティビティ・プログラムと通信する少なくとも1つのアクティビティ・プログラム・インターフェース(API)を含み、少なくとも1つのアクティビティ・プログラム・インターフェースが、少なくとも1つのアクティビティ・プログラムとアプリケーション・ゲートウェイとの間で通信する、上記(11)のコンピュータ・システム。

(15) ウェブ・サーバ・アプリケーションが、複数のウェブ・ブラウザのうちの1つから渡された認証データから、選択されたウェブ・ブラウザがウェブ・サーバ・アプリケーションへのアクセスを許可されるかどうかを判定する認証機構を含み、ウェブ・サーバが、複数のウェブ・ブラウザから受け取るデータ及びアプリケーション・ゲートウェイから受け取るデータを処理する上記(11)のコンピュータ・システム。

(16) ウェブ・ブラウザが、複数のCPUのうちの少なくとも1つによってクライアント・ワークステーショ

ン上で実行される、上記(11)のコンピュータ・システム。

(17) ウェブ・サーバ・アプリケーションが、複数のCPUのうちの少なくとも1つによって、ウェブ・サーバ・コンピュータ上で実行される、上記(11)のコンピュータ・システム。

(18) アプリケーション・ゲートウェイが、複数のCPUのうちの少なくとも1つによって、ウェブ・サーバ・コンピュータ上で実行される、上記(11)のコンピュータ・システム。

(19) アプリケーション・ゲートウェイが、複数のCPUのうちの少なくとも1つによって、第1のコンピュータ上で実行される、上記(11)のコンピュータ・システム。

(20) ソフトウェア・アプリケーションが、複数のCPUのうちの少なくとも1つによって、第2のコンピュータ上で実行される、上記(11)のコンピュータ・システム。

(21) アプリケーション・ゲートウェイが、複数のCPUのうちの少なくとも1つによって、第2のコンピュータ上で実行される、上記(11)のコンピュータ・システム。

(22) 複数の中央処理装置(CPU)を提供するステップと、複数のCPUに結合されたメモリを提供するステップと、複数のCPUのうちの少なくとも1つによって、メモリ内に常駐する複数のウェブ・ブラウザのうちの少なくとも1つを実行するステップと、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、ウェブ・サーバ・アプリケーションを提供するステップと、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、ソフトウェア・アプリケーションを提供するステップと、メモリ内に常駐し、複数のCPUのうちの少なくとも1つによって実行される、アプリケーション・ゲートウェイを提供するステップと、認証データ及び環境データをウェブ・サーバ・アプリケーションに送ることによって、複数のウェブ・ブラウザのうちの選択された1つが、ソフトウェア・アプリケーションへのアクセスを開始するステップと、認証データが選択されたウェブ・ブラウザにウェブ・サーバ・アプリケーションへのアクセスを許可する場合に、環境データを処理するステップと、処理された環境データをアプリケーション・ゲートウェイに出力するステップと、選択されたウェブ・ブラウザと、ソフトウェア・アプリケーションによって実行される所望のプロセスとに対応する識別子を生成するステップと、メモリ内に常駐し、少なくとも1つのCPUによって実行され、ウェブ・ブラウザからユーザ入力を受け取り、受け取った入力に対応するソフトウェア・アプリケーションの認証パラメータを取り出す、セキュリティ機構を提供するステップと、メモリ内に常駐し、少なくとも1つの

CPUによって実行され、ウェブ・ブラウザとソフトウェア・アプリケーションとの間で変数及びテンプレートを送受する、インターフェース機構を提供するステップと、メモリ内に常駐し、少なくとも1つのCPUによって実行され、会話が再開されてソフトウェア・アプリケーションによって所望のプロセスが実行される時に状態データを取り出すことができるように、会話が中止される時にウェブ・ブラウザとソフトウェア・アプリケーションとの間の会話に関連する状態及び会話識別子を記憶する、切断機構を提供するステップと、所望のプロセスを実行した結果を、識別子を有するアプリケーション・ゲートウェイに返すステップと、識別子に基づいて、複数のブラウザのうちのどれに結果を送らなければならないかを決定するステップと、アプリケーション・ゲートウェイからウェブ・サーバ・アプリケーションに結果を送るステップと、識別子に対応する選択された1つのウェブ・ブラウザに、サーバから結果を送るステップとを含む、ワールド・ワイド・ウェブ上でウェブ・ブラウザとソフトウェア・アプリケーションとの間で通信するための共通ユーザ・インターフェースを提供するための、コンピュータ実施される方法。

(23) ソフトウェア・アプリケーションが、プロセス・エンジニアリング・ソフトウェア・アプリケーションである、上記(22)の方法。

(24) ウェブ・ブラウザを走行させるクライアント・ワークステーションと、ウェブ・サーバ・アプリケーションを走行させるウェブ・サーバ・コンピュータと、アプリケーション・ゲートウェイを走行させる第1コンピュータと、ソフトウェア・アプリケーションを走行させる第2コンピュータと、ウェブ・ブラウザとウェブ・サーバ・アプリケーションとの間でデータを伝送できるようにする、ウェブ・ブラウザとウェブ・サーバ・アプリケーションとの間の通信機構と、ウェブ・サーバ・アプリケーションとアプリケーション・ゲートウェイとの間でデータを伝送できるようにする、ウェブ・サーバ・アプリケーションとアプリケーション・ゲートウェイとの間の通信機構と、アプリケーション・ゲートウェイとソフトウェア・アプリケーションとの間でデータを伝送できるようにする、アプリケーション・ゲートウェイとソフトウェア・アプリケーションとの間の通信機構と、ウェブ・ブラウザとソフトウェア・アプリケーションとの間で変数及びテンプレートを送受する、インターフェース機構と、ウェブ・ブラウザとソフトウェア・アプリケーションとの間に結合され、ウェブ・ブラウザとソフトウェア・アプリケーションとの間のインターフェースを提供する、セキュリティ機構と、会話が再開されてソフトウェア・アプリケーションによって所望のプロセスが実行される時に状態データを取り出すことができるように、会話が中止される時にウェブ・ブラウザとソフトウェア・アプリケーションとの間の会話に関連する状態及

び会話識別子を記憶する、切断機構と、ウェブ・ブラウザがワールド・ワイド・ウェブ上でソフトウェア・アプリケーションと通信できるようにする、複数のアプリケーション・プログラミング・インターフェースとを含む、ワールド・ワイド・ウェブ上でウェブ・ブラウザとソフトウェア・アプリケーションの間で通信するための共通ユーザ・インターフェースを提供するためのシステム。

(25) ソフトウェア・アプリケーションが、プロセス・エンジニアリング・ソフトウェア・アプリケーション 10 である、上記(24)のシステム。

(26) ウェブ・サーバ・コンピュータが、第1コンピュータを含む、上記(24)のシステム。

(27) 第1コンピュータが、第2コンピュータを含む、上記(24)のシステム。

【図面の簡単な説明】

【図1】本発明の好ましい実施例のブロック図である。

【図2】クライアント・ワークステーションとウェブ・サーバの間のトランザクションを示すブロック図である。

【図3】標準ウェブ・ブラウザからワールド・ワイド・ウェブを介してソフトウェア・アプリケーションにアクセスできるようにする、本発明の好ましい実施例によるシステムのブロック図である。

【図4】ワールド・ワイド・ウェブを介するFlowMarkワークフロー・アプリケーション・ソフトウェアへのアクセスにさらに適合された、図3のシステムの詳細なブロック図である。

【図5】共通ユーザ・インターフェースの特徴の一部を示す、本発明の好ましい実施例による方法の流れ図である。 30

【図6】図5のセキュリティ検査機能の処理流れ図である。

【図7】図5のテンプレート/HTML変数機能の処理流れ図である。

【図8】図5のRESUME API機能及びDISCONNECT API機能の処理流れ図である。

【図9】ユーザ・ライブラリの一部を表す表である。

【図10】複数ユーザー環境に拡張された時の本発明の好ましい実施例のブロック図である。 40

【図11】本発明の好ましい実施例による、自動車レンタル予約フォームの生成に使用されるHTMLコードの例を示す図である。

【図12】本発明の好ましい実施例に従って、ウェブ・ブラウザからのユーザ要求によって生成されるデータ・ストリームの例を示す図である。

【図13】本発明の好ましい実施例による、予約確認テンプレートの生成に使用されるHTMLコードの例を示す図である。

【図14】本発明の好ましい実施例に従って、CGIに 50

よって処理された後の図10のHTMLコードの例を示す図である。

【図15】本発明の好ましい実施例に従ってウェブ・ブラウザが書式化したデータの例を示す図である。

【図16】本発明の好ましい実施例による、自動車レンタル予約代理人用のウェブ・ページの生成に使用されるHTMLコードの例を示す図である。

【図17】本発明の好ましい実施例に従ってウェブ・サーバが生成したデータの例を示す図である。

【図18】本発明の好ましい実施例による、レンタ・カー代理人作業リストの生成に使用されるHTMLコードの例を示す図である。

【図19】本発明の好ましい実施例による、レンタ・カー代理人作業リストの生成に使用されるHTMLコードの例を示す図である。

【図20】本発明の好ましい実施例に従ってCGIによって処理された後の図15のHTMLコードの例を示す図である。

【図21】本発明の好ましい実施例に従ってCGIによって処理された後の図15のHTMLコードの例を示す図である。 20

【図22】本発明の好ましい実施例に従ってCGIによって処理された後の図15のHTMLコードの例を示す図である。

【図23】本発明の好ましい実施例による、ウェブ・ブラウザからCIGによって受信されたデータ・ストリームの例を示す図である。

【図24】本発明の好ましい実施例に従って自動車使用可能性ページを生成するのに使用されるHTMLコードの例を示す図である。

【図25】本発明の好ましい実施例に従ってCGIによって処理された後の図18のHTMLコードの例を示す図である。

【図26】本発明の好ましい実施例を使用するソフトウェア・アプリケーションとのWWWトランザクションを記述した、処理モデル図である。

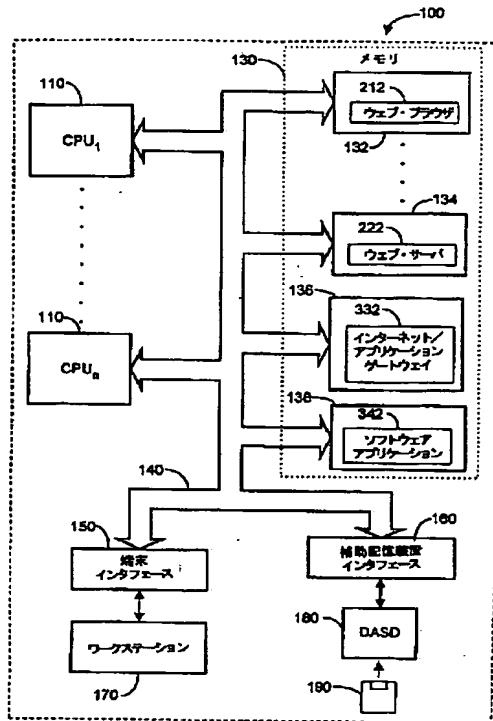
【符号の説明】

- 100 システム
- 110 中央処理装置(CPU)
- 130 メモリ
- 132 記憶位置
- 134 記憶位置
- 136 記憶位置
- 138 記憶位置
- 140 バス
- 150 端末インターフェース
- 160 補助記憶装置インターフェース
- 170 ワークステーション
- 180 直接アクセス記憶装置(DASD)
- 190 フロッピー・ディスク

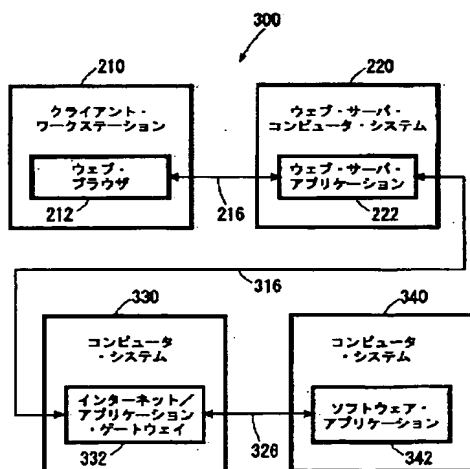
- 212 ウェブ・ブラウザ
 222 ウェブ・サーバ・アプリケーション
 332 インターネット/アプリケーション・ゲートウェイ

- *エイ(ゲートウェイ)
 342 ソフトウェア・アプリケーション

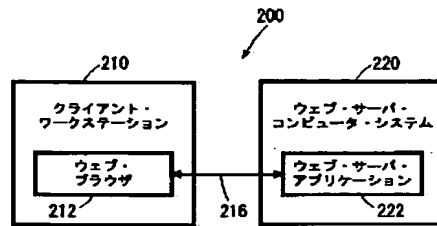
【図1】



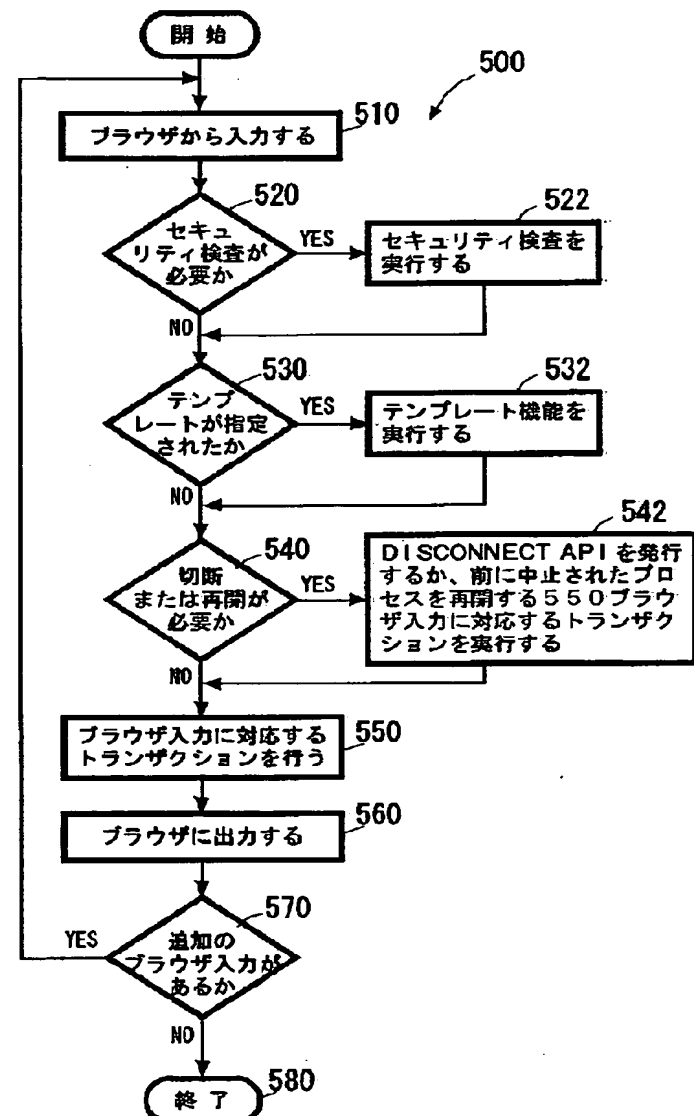
【図3】



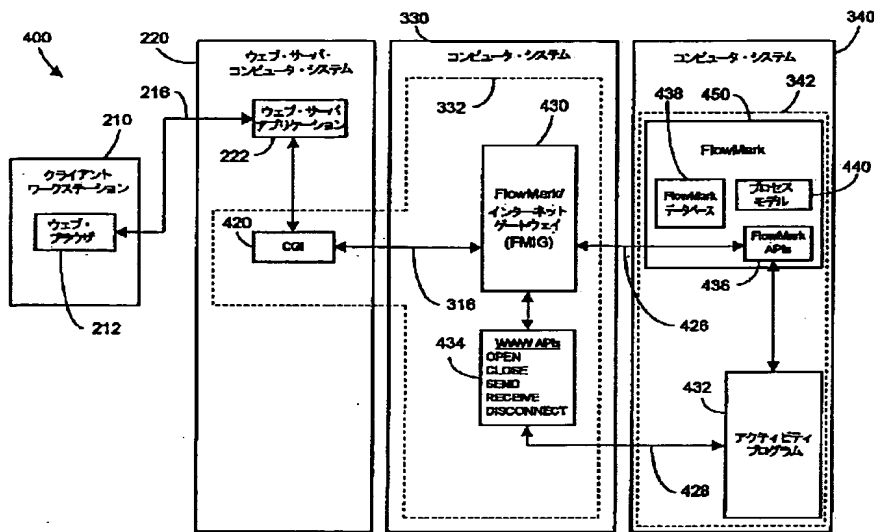
【図2】



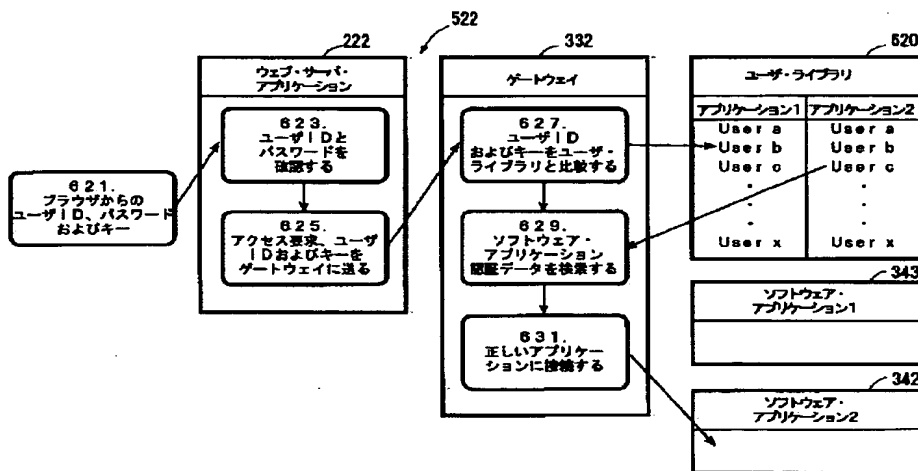
【図5】



【図4】



【図6】



【図9】

900

データベース サーバ	ユーザ名	パスワード	データベース サーバ	データベース
User a	Joe Brown	Secret	CorpServer	CorpInfo
User b				
User c				
User x				

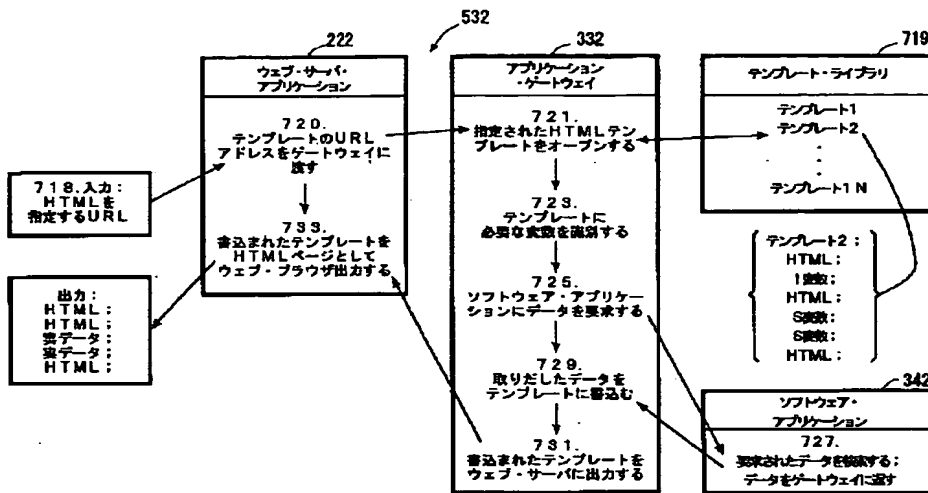
【図13】

```

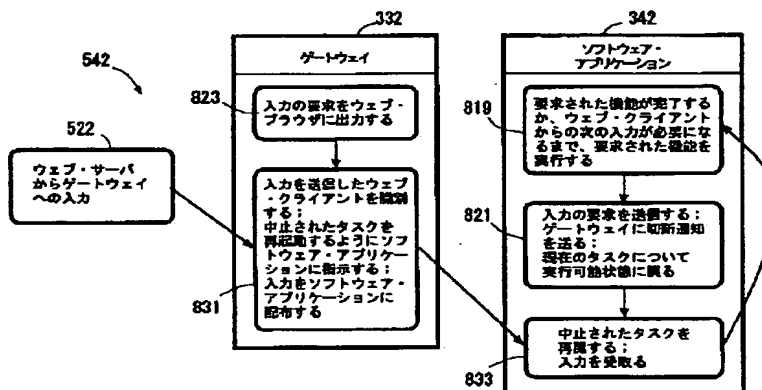
<html>
<title>Cars Around the World</title>
<FORM ACTION="/cgi-prothemp/cgluser" METHOD="POST">
<div style="text-align: center;>Customer's Reservation Number</div>
<div>
<input type="text" value="" />
</div>
<div>
<input type="submit" value="OK" />
<input type="hidden" name="wf-cgl-submit" value="13" />
</div>
</html>

```

【図7】



【図8】



【図12】

```

membno=1234&lname=Doe&fname=John&mi=E&origcity=Nevada&origstate=IA&startdate=09%2F23%2F98&days=4&cartype=2&wf-cgi-submit=2&wf-api-proc-template=www_Reservation_Request&wf-fmig-key=webfmcust&wf-cgi-html=%2Fexm%2Fsmp%2Fexmp5srk.htm

```

【図15】

```

wf-cgi-submit=13&wf-fmig-handle=007777775F5285736572768174696F8wf-fmig-key=webfmcust

```

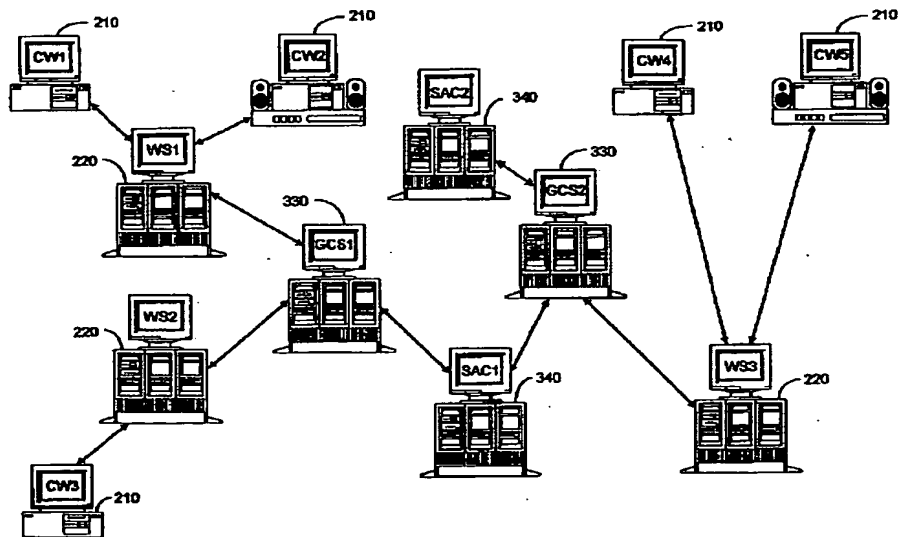
【図17】

```

wf-fmig-key=webfmagent&wf-cgi-html=%2Fexm%2Fhtml%2Fexmp5ewi.htm&wf-cgisubmit=0

```

【図10】



【図11】

```

<html>
<title>Cars Around the World</title>
<FORM ACTION="/cgi-proc/exmp5cgi.exe" METHOD="POST">
<H3>Rental Reservation</H3>
<HR>
Enter the following information and then press submit:
<HR>
<p>Member number: <INPUT TYPE="text" NAME="membno" SIZE=4>
<p>Name:      Last: <INPUT TYPE="text" NAME="lname" SIZE=15>
               First: <INPUT TYPE="text" NAME="fname" SIZE=15>
               MI: <INPUT TYPE="text" NAME="mi" SIZE=1>
<HR>
<p>Origin City: <INPUT TYPE="text" NAME="origcity" SIZE=15>
               State: <INPUT TYPE="text" NAME="origstate" SIZE=2>
<p>Start Date (MM/DD.YY): <INPUT TYPE="text" NAME="startdate" SIZE=8>
               Number of days <INPUT TYPE="text" NAME="days" SIZE=3>
<p>Select Car Preference:
      <INPUT TYPE="radio" NAME="cartype" VALUE=1> Compact
      <INPUT TYPE="radio" NAME="cartype" CHECKED VALUE=2> Mid Size
      <INPUT TYPE="radio" NAME="cartype" VALUE=3> Full Size
      <INPUT TYPE="radio" NAME="cartype" VALUE=4> Luxury
<HR>
<p><INPUT TYPE="submit" VALUE="Submit">
<p><INPUT TYPE="hidden" NAME="wf-cgi-submit" VALUE="2">
<p><INPUT TYPE="hidden" NAME="wf-apl-proc-template" VALUE="www_Reservation Request">
<INPUT TYPE="hidden" NAME="wf-fmlg-key" VALUE="webfmcust">
<INPUT TYPE="hidden" NAME="wf-cgi-html" VALUE="/exm/smp/exmp5srk.htm">
</form>
</html>

```

【図23】

```

wf-apl-item="TTTTT75F5363686564756C855F5265736572768174696F6E00777775F52857385
72768174696F6E5F526571756573745F383500454D4147454E54007765626167656E7400776562
666D6167656E74"&wf-cgi-submit=3&wf-cgi-html=%2Fexm%2Fhtml%2Fexmp5ewl.htm&wf-fmlg-
key=webfmagent

```

【図14】

```

<html>
<title>Cars Around the World</title>
<body>
<FORM ACTION="/cgi-prot/exmp5cgi.exe" METHOD="POST">
<H3><CENTER>Customer's Reservation Number</CENTER></H3>
<HR>
<p>Your reservation number is 4411. </p>
<HR>
<p><INPUT TYPE="submit" VALUE="OK">
<p><INPUT TYPE="hidden" NAME="wf-cgi-submit" VALUE="13">
<INPUT TYPE="hidden" NAME="wf-fmig-handle" VALUE="007777775F52657365727661746868F">
<INPUT TYPE="hidden" NAME="wf-fmig-key" VALUE="webfmoust">
</form>
</html>

```

【図16】

```

<html>
<FORM ACTION="/cgi-prot/exmp5cgi.exe" METHOD="POST">
<HEAD>
<title>IBM Internet Connection for FlowMark home page</title>
<HEAD>
<BODY>
<h3>
IBM Internet Connection for FlowMark home page
</h3>
<hr>
<h4>
Enter key, select next screen, then press the submit button.
</h4>
<strong>
Key:
</strong>
<INPUT TYPE="text" NAME="wf-fmig-key" SIZE=64>
<br>
<INPUT TYPE="radio" NAME="wf-cgi-html" VALUE="/exn/html/exmp5ewl.htm" CHECKED>
<IMG SRC="/exn/cons/exmp5ewl.gif" ALIGN=MIDDLE HSPACE=5>Work with Work Items
<br>
<INPUT TYPE="radio" NAME="wf-cgi-html" VALUE="/exn/html/exmp5epi.htm">
<IMG SRC="/exn/cons/exmp5epi.gif" ALIGN=MIDDLE HSPACE=5>Work with Process Instances
<br>
<INPUT TYPE="radio" NAME="wf-cgi-html" VALUE="/exn/html/exmp5ept.htm">
<IMG SRC="/exn/cons/exmp5ept.gif" ALIGN=MIDDLE HSPACE=5>Work with Process Templates

<!-- Start Submit Button & hidden variables ----->
<hr>
<INPUT TYPE="submit" VALUE="Submit">
<INPUT TYPE="hidden" NAME="wf-cgi-submit" VALUE="0">

<!-- End Submit Button & hidden variables ----->
<br>
<b>
[
<a href="/exn/docs/exmp5d20.htm">Help |
<a href="http://www.ibm.com/">IBM home page</a>> |
<a href="http://www.software.ibm.com/ad/flowmark/exmn0mst.htm">FlowMark home page</a>
]

</FORM>
</BODY>

```

【図18】

```

<html>
<FORM ACTION="/cgi-prot/exmp5cgi.exe" METHOD="POST">
<HEAD>
<title>FlowMark - Work Items</title>
</HEAD>
<BODY>

<h4>Select a work item:</h4>
<TABLE BORDER=1>
<tr>
<th COLSPAN=2>Description
<th>Status
<th>Activity name
<th>Process
<th>Program
<th>Received<br>date & time
<th>Priority
<th>Category
<!-- "wf-cgi-rbegin" -->
<tr>
<td ALIGN=CENTER VALIGN=MIDDLE><INPUT TYPE="radio"
NAME="wf-apl-item" VALUE="wf-apl-item-id">
<td NOWRAP><!-- "wf-apl-item-descrp" -->
<td><!-- "wf-apl-item-state" -->
<td><!-- "wf-apl-item-name" -->
<td><!-- "wf-apl-item-procinst" -->
<td><!-- "wf-apl-item-impl" -->
<td ALIGN=CENTER><!-- "wf-apl-item-starttime" -->
<td ALIGN=CENTER><!-- "wf-apl-item-priority" -->
<td><!-- "wf-apl-item-category" -->
<!-- "wf-cgi-rend" -->
</TABLE>

<tr>
<td>
<h4>Next, select which action you want to perform and
<tr>
press the button below.
</h4>

<TABLE BORDER=0>
<tr>
<td>
<td>
<INPUT TYPE="radio" NAME="wf-cgi-submit" VALUE="3" CHECKED>
Start work item
<tr>
<td>
<td>

```

【図19】

```

<INPUT TYPE="radio" NAME="wf-cgi-submit" VALUE="0">
Refresh list
</TABLE>
<!-- Start Submit Button & hidden variables ----->
<hr>
<INPUT TYPE="submit" VALUE="Perform Action">
<INPUT TYPE="hidden" NAME="wf-cgi-html" VALUE="/exn/html/exmp5ewl.htm">
<!-- End Submit Button & hidden variables ----->
<hr>
<a href="/exn/html/exmp5ehp.htm">

Return to home page
</a>
<a href="/exn/html/exmp5d20.htm">

Help
</a>
<hr>
<TABLE BORDER=1>
<tr>
<td>FlowMark User
<td>FlowMark Database
<td>FlowMark Server
<td>Date & Time
<tr>
<td><!-- "wf-api-fmuser" -->
<td><!-- "wf-api-fmdb" -->
<td><!-- "wf-api-fmserver" -->
<td><!-- "wf-api-datetime" -->
</table>
</FORM>
</BODY>
</HTML>

```

【図22】

```

<TABLE BORDER=1>
<tr>
<td>FlowMark User
<td>FlowMark Database
<td>FlowMark Server
<td>Date & Time
<tr>
<td>FMAGENT
<td>WWWDB
<td>WWWSRV
<td>08:27:33 09/05/96
</table>
<INPUT TYPE="hidden" NAME="wf-fmig-key" VALUE="webfmagent">
</FORM>
</BODY>
</HTML>

```


【図20】

```

<html>
<FONT SIZE=3>
<FORM ACTION="/cgi-pro/exmp5cgi.exe" METHOD="POST">
<HEAD>
<title>FlowMark - Work Items</title>
</HEAD>
<BODY>

<h4>Select a work item:</h4>
<TABLE BORDER=1>
<tr>
<th COLSPAN=2>Description
<th>Status
<th>Activity name
<th>Process
<th>Program
<th>Received<br>date & time
<th>Priority
<th>Category
<!-- "wf-cgi-rbegin" -->
<tr>
<td ALIGN=CENTER VALIGN=MIDDLE><INPUT TYPE="radio"
NAME="wf-apt-item"
VALUE="7777775F5363686584756C855F5265736572786174686F8E00777775F52657365727861
74686F8E5F526571756573745F383500454D4147454E54007765826167656E7400776562686D616
7656E74">
<td> NOWRAP>Schedule car
<td>Ready
<td>www_Schedule_Reservation
<td>www_Reservation Request_85
<td>Schedule_Reservation
<td ALIGN=CENTER>09-03-1996 02:55:11PM
<td ALIGN=CENTER>4004128
<td>
</tr>
<tr>
<td>ALIGN=CENTER VALIGN=MIDDLE><INPUT TYPE="radio"
NAME="wf-apt-item"
VALUE="7777775F5363686584756C855F5265736572786174686F8E00777775F52657365727861
74686F8E5F526571756573745F383500454D4147454E54007765826167656E7400776562686D616
7656E74">
<td> NOWRAP>Schedule car
<td>Ready
<td>www_Schedule_Reservation
<td>www_Reservation Request_84
<td>Schedule_Reservation
<td ALIGN=CENTER>09-03-1996 01:48:48PM
<td ALIGN=CENTER>4004128
<td>
</tr>
<tr>
<td>ALIGN=CENTER VALIGN=MIDDLE><INPUT TYPE="radio"

```

【図21】

```

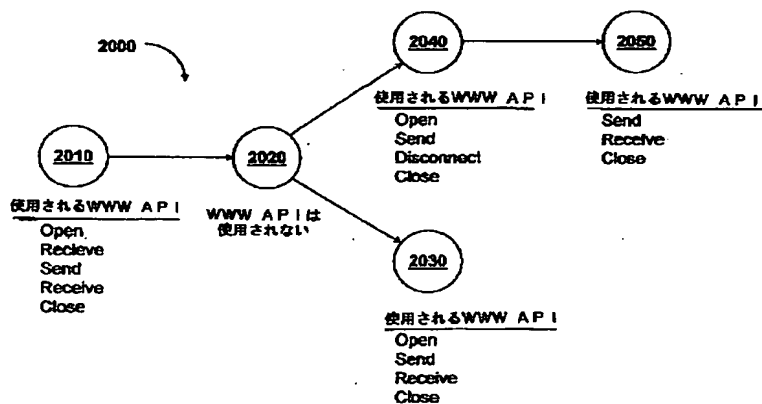
NAME="wf-api-item"
VALUE="7777775F5363686564758C655F5265736572766174686F6E00777775F52657365727661
74686F6E5F526571756573745F383500454D4147454E54007785826167656E7400776582686D616
7656E74">
    <td> NOWRAP>Schedule car.
    <td>Ready
    <td>www_Schedule_Reservation
    <td>www_Reservation Request_86
    <td>Schedule_Reservation
    <td ALIGN="CENTER">09-03-1996 03:01:15PM
    <td ALIGN="CENTER">4004128
    <td>
<!-- "wf-cgi-rend" -->
</TABLE>
<hr>
<h4>Next, select which action you want to perform and
<br>
press the button below.
</h4>
<TABLE BORDER=0>
<tr>
<td>
<INPUT TYPE="radio" NAME="wf-cgi-submit" VALUE="3" CHECKED>
Start work item
</td>
<td>
<INPUT TYPE="radio" NAME="wf-cgi-submit" VALUE="0">
Refresh list
</td>
</tr>
</TABLE>
<!-- Start Submit Button & hidden variables ----->
<hr>
<INPUT TYPE="submit" VALUE="Perform Action">
<INPUT TYPE="hidden" NAME="wf-cgi-hnm" VALUE="/exn/html/exmp5ewi.htm">
<!-- End Submit Button & hidden variables ----->
<hr>
<a href="/exn/html/exmp5ehp.htm">

Return to home page
</a>
<a href="/exn/html/exmp5d20.htm">

Help
</a>
<hr>

```

【図26】



【図24】

```

<HTML>
<FORM ACTION="/cgi-pro/exp5cgi.exe" METHOD="POST">
<HEAD>
<TITLE>Customer Information</TITLE>
<B>Customer Reservation Information</B>
<table border=1>
<tr><td>Name</td><td>Member Number</td><td>Resv Number</td>
<tr><td><!-- "wf-act-name" --></td>
<td align=right>"wf-act-membno"</td>
<td align=right>"wf-act-resvno"</td>
</table>
<p><!-- "wf-act-outmsg" -->
<HR>
<B>Move An Available Ca</B>
<p><INPUT TYPE="radio" NAME=cars CHECKED> <!-- "wf-act-car1" -->
<p><INPUT TYPE="radio" NAME=cars > <!-- "wf-act-car2" -->
<p><INPUT TYPE="radio" NAME=cars > <!-- "wf-act-car3" -->
<p><INPUT TYPE="radio" NAME=cars > <!-- "wf-act-car4" -->
<HR>
<INPUT TYPE="submit" NAME="move" VALUE="Move Car">
<INPUT TYPE="submit" NAME="cancel" VALUE="Cancel">
<INPUT TYPE="hidden" NAME="wf-cgi-submit" VALUE="13">
</FORM>
</HTML>

```

【図25】

```

<HTML>
<FORM ACTION="/cgi-pro/exp5cgi.exe" METHOD="POST">
<HEAD>
<TITLE>Customer Information</TITLE>
<B>Customer Reservation Information</B>
<table border=1>
<tr><td>Name</td><td>Member Number</td><td>Resv Number</td>
<tr><td><John E. Doe</td>
<td align=right>1234</td>
<td align=right>4412</td>
</table>
<p>Reservation Date: <B>09/24/96</B> Days Requested:
<B>4</B><p>Car Type Requested: <B>Luxury</B><p>Origin: <B>Nevada, IA</b>
<HR>
<B>Move An Available Ca</B>
<p><INPUT TYPE="radio" NAME=cars CHECKED> XYD123 Pontiac Grand Am
<p><INPUT TYPE="radio" NAME=cars > Chevy Camaro Z28
<p><INPUT TYPE="radio" NAME=cars > Chevrolet Lumina
<p><INPUT TYPE="radio" NAME=cars > Oldmobile Cutlass Supreme
<HR>
<INPUT TYPE="submit" NAME="move" VALUE="Move Car">
<INPUT TYPE="submit" NAME="cancel" VALUE="Cancel">
<INPUT TYPE="hidden" NAME="wf-cgi-submit" VALUE="13">
<INPUT TYPE="hidden" NAME="wf-fmig-handle"
VALUE="7777775F4361725F356E617661696C61626C85007777775F5265736572766174696F6E5F
526571756573745F3837">
<INPUT TYPE="hidden" NAME="wf-fmig-key" VALUE="webfmagent">
</FORM>
</HTML>

```

フロントページの続き

- (72)発明者 ケニス・エドガー・ブラウン
アメリカ合衆国55901 ミネソタ州ロチェ
スターサーティーサード・ストリート ノ
ースウェスト 937
- (72)発明者 バーネル・ジェームズ・ダイクス
アメリカ合衆国55920 ミネソタ州バイロ
ン サード・アベニュー ノースウェスト
720
- (72)発明者 エリック・ドゥエーン・リンダバーグ
アメリカ合衆国55906 ミネソタ州ロチェ
スターリバーサイド・レーン ノースイー
スト 2685

- (72)発明者 ダイアン・イレイン・オルソン
アメリカ合衆国55901 ミネソタ州ロチェ
スターナインティーンズ・アベニュー ノ
ースウェスト 3910 ナンバー17
- (72)発明者 ジェフリー・エドワード・セルデン
アメリカ合衆国32250 フロリダ州ジャク
ソンヴィル・ビーチ ペンマン・ロード
920
- (72)発明者 デヴォン・ダニエル・スナイダー
アメリカ合衆国55906 ミネソタ州ロチェ
スターグレンデール・ヒルズ・ドライブ
ノースイースト 1201
- (72)発明者 ジェームズ・オリン・ワルツ
アメリカ合衆国55901 ミネソタ州ロチェ
スターナインティーンズ・ストリート ノ
ースウェスト 408